SSL Offload and Acceleration

http://docs.citrix.com/content/docs/en-us/netscaler/11-1/ssl.html
Dec. 08, 2014

SSL Offload and Acceleration

A CitrixÂ® NetScalerÂ® appliance configured for SSL acceleration transparently accelerates SSL transactions by offloading SSL processing from the server. To configure SSL offloading, you configure a virtual server to intercept and process SSL transactions, and send the decrypted traffic to the server (unless you configure end-to-end encryption, in which case the traffic is re-encrypted). Upon receiving the response from the server, the appliance completes the secure transaction with the client. From the client's perspective, the transaction seems to be directly with the server. A NetScaler configured for SSL acceleration also performs other configured functions, such as load balancing.

Configuring SSL offloading requires an SSL certificate and key pair, which you must obtain if you do not already have an SSL certificate. Other SSL-related tasks that you might need to perform include managing certificates, managing certificate revocation lists, configuring client authentication, and managing SSL actions and policies.

A non-FIPS NetScaler appliance stores the serverâ€™s private key on the hard disk. On a FIPS appliance, the key is stored in a cryptographic module known as a hardware security module (HSM). Only the MPX 9700/10500/12500/15500 appliances support a FIPS card, so other NetScaler models cannot be equipped with an HSM.

Beginning with release 10.5, build 52.1115.e, all NetScaler appliances that do not support a FIPS card (including virtual appliances) support the Thales nShieldÂ® Connect external HSM. (MPX 9700/10500/12500/15500 appliances do not support an external HSM.)

Note: FIPS-related options for some of the SSL configuration procedures described in this document are specific to a FIPS-enabled NetScaler.

Configuring SSL Offloading

To configure SSL offloading, you must enable SSL processing on the NetScaler appliance and configure an SSL based virtual server that will intercept SSL traffic, decrypt the traffic, and forward it to a service that is bound to the virtual server. To secure time-sensitive traffic, such as media streaming, you can configure a DTLS virtual server.To enable SSL offloading, you must import a valid certificate and key and bind the pair to the virtual server.

To configure SSL offloading, see the following sections:

Enabling SSL Processing

To process SSL traffic, you must enable SSL processing. You can configure SSL based entities, such as virtual servers and services, without enabling SSL processing, but they will not work until SSL processing is enabled.

To enable SSL processing by using the command line interface

At the command prompt, type:

- enable ns feature `ssl`
- show ns feature

Example

```
> enable ns feature SSL
 Done
> show ns feature

        Feature                        Acronym              Status
        -------                        -------              ------
 1)     Web Logging                    WL                   OFF
 2)     Surge Protection               SP                   ON
 3)     Load Balancing                 LB                   ON
 .
 .
 .
 9)     SSL Offloading                 SSL                  ON
 .
 .
 .
 24)    NetScaler Push                 push                 OFF
 Done
```

To enable SSL processing by using the configuration utility

Navigate to System > Settings and, in the Modes and Features group, select Configure Basic Features, and select SSL Offloading.

Configuring Services

On the NetScaler appliance, a service represents a physical server or an application on a physical server. Once configured, services are in the disabled state until the appliance can reach the physical server on the network and monitor its status.

To add a service by using the command line interface

At the command prompt, type the following commands to add a service and verify the configuration:

- add service <name> (<IP> | <serverName>) <serviceType> <port>
- show service <serviceName>

Example

```
> add service SSL1 192.168.0.12 SSL 443

 Done

> sh service SSL1

        SSL1 (192.168.0.12:443) - SSL

        State: DOWN

        Last state change was at Thu Oct  6 17:15:41 2016

        Time since last state change: 0 days, 00:00:07.990

        Server Name: 192.168.0.12

        Server ID : None        Monitor Threshold : 0

        Max Conn: 0     Max Req: 0      Max Bandwidth: 0 kbits

        Use Source IP: NO

        Client Keepalive(CKA): NO

        Access Down Service: NO

        TCP Buffering(TCPB): NO

        HTTP Compression(CMP): NO

        Idle timeout: Client: 180 sec   Server: 360 sec

        Client IP: DISABLED

        Cacheable: NO

        SC: OFF

        SP: ON

        Down state flush: ENABLED

        Monitor Connection Close : NONE

        Appflow logging: ENABLED

        Process Local: DISABLED

        Traffic Domain: 0


1)      Monitor Name: tcp-default

                State: DOWN     Weight: 1       Passive: 0

                Probes: 2       Failed [Total: 1 Current: 1]

                Last response: Failure - Time out during TCP connection
establishment stage

                Response Time: 0.0 millisec

 Done
```

To modify or remove a service by using the command line interface

To modify a service, use the set service command, which is just like using the add service command, except that you enter the name of an existing service. To remove a service, use the rm service command, which accepts only the <name> argument.

To configure a service by using the configuration utility

Navigate to Traffic Management > Load Balancing > Services, create a service, and specify the protocol as SSL.

Configuring an SSL-Based Virtual Server

Secure sessions require establishing a connection between the client and an SSL-based virtual server on the NetScaler appliance. The SSL virtual server intercepts SSL traffic, decrypts it and processes it before sending it to services that are bound to the virtual server.

Note: The SSL virtual server is marked as down on the NetScaler appliance until a valid certificate / key pair and at least one service are bound to it. An SSL based virtual server is a load balancing virtual server of protocol type SSL or SSL_TCP. The load balancing feature must be enabled on the NetScaler.

To add an SSL-based virtual server by using the command line interface

At the command prompt, type the following commands to create an SSL-based virtual server and verify the configuration:

- add lb vserver <name> (serviceType) <IPAddress> <port>
- show lb vserver <name>

Example

```
> add lb vserver vssl  SSL 10.102.29.133 443
 Done
> show ssl vserver vssl

        Advanced SSL configuration for VServer vssl:
        DH: DISABLED
        Ephemeral RSA: ENABLED          Refresh Count: 0
        Session Reuse: ENABLED          Timeout: 120 seconds
        Cipher Redirect: DISABLED
        SSLv2 Redirect: DISABLED
        ClearText Port: 0
        Client Auth: DISABLED
        SSL Redirect: DISABLED
        Non FIPS Ciphers: DISABLED
        SSLv2: DISABLED SSLv3: ENABLED  TLSv1: ENABLED

1)      Cipher Name: DEFAULT
        Description: Predefined Cipher Alias
 Done
```

To modify or remove an SSL-based virtual server by using the command line interface

To modify the load balancing properties of an SSL virtual server, use the set lb vserver command, which is just like using the add lb vserver command, except that you enter the name of an existing vserver. To modify the SSL properties of an SSL-based virtual server, use the set ssl vserver command. For more information, see Customizing the SSL Configuration.

To remove an SSL virtual server, use the rm lb vserver command, which accepts only the <name> argument.

To configure an SSL-based virtual server by using the configuration utility

Navigate to Traffic Management > Load Balancing > Virtual Servers, create a virtual server, and specify the protocol as SSL.

Configuring an HTTPS Virtual Server to accept HTTP Traffic

A user might attempt to access a secure web site by sending an HTTP request. You can drop such requests or redirect the request to the secure web site. In earlier releases, to redirect the request to the secure web site, you were required to do the following:

- Add HTTP and HTTPS virtual servers with the same IP address but different ports.
- Add a responder action that redirects all traffic to the HTTPS virtual server.
- Add a responder policy specifying the above action, and bind the policy to the HTTP virtual server.

From release 11.1, you can configure an HTTPS virtual server to also process all HTTP traffic. That is, if HTTP traffic is received on the HTTPS virtual server, the appliance internally prepends "**https://**" to the incoming URL or redirects the traffic to another HTTPS URL, depending on the option configured.

To achieve this, two new parameters, -**httpsRedirectUrl** and -**redirectFromPort** are added to the "add lb vserver" command.

- **redirectFromPort**: All HTTP traffic received on this port is prefixed with https:// in the URL and redirected.
- **httpsRedirectUrl**: All HTTP traffic received on the port specified in the **-redirectFromPort** parameter is redirected to this URL. For example, all HTTPS traffic received on http://www.example.com is redirected to https://www.sample.com.

To configure HTTP to HTTPS redirect by using the NetScaler command line

At the command prompt, type:

add lb vserver <name> <serviceType> -**redirectFromPort** <port **|** *> -**httpsRedirectUrl** <URL>

Example

```
> add lbvserver lbvip2 SSL 1.2.1.2 443 -redirectFromPort 80 -httpsRedirectUrl
https://www.example.com

 Done
```

To configure HTTP to HTTPS redirect by using the NetScaler GUI

1. Navigate to **Traffic Management** > **Load Balancing** > **Virtual Servers**.
2. Add a virtual server of type **SSL** and click **OK**.
3. Edit **Basic Settings**, click **More**, and add values for **Redirect From Port** and **HTTPS Redirect URL**.

Binding Services to the SSL-Based Virtual Server

For the NetScaler appliance to forward decrypted SSL data to servers in the network, services representing these physical servers must be bound to the virtual server that receives the SSL data.

Because the link between the NetScaler and the physical server is typically secure, data transfer between the appliance and the physical server does not have to be encrypted. However, you can provide end-to end-encryption by encrypting data transfer between the NetScaler and the server. For details, see Configuring SSL Offloading with End-to-End Encryption.

Note: The Load Balancing feature should be enabled on the NetScaler appliance before you bind services to the SSL based virtual server.

To bind a service to a virtual server by using the command line interface

At the command prompt, type the following commands to bind the service to the virtual server and verify the configuration:

- bind lb vserver <name> <serviceName>
- show lb vserver <name>

Example

```
> bind lb vserver vssl ssl1
 Done
> show lb vserver vssl
        vssl (10.102.29.133:443) - SSL  Type: ADDRESS
        State: DOWN[Certkey not bound]
        Last state change was at Thu Nov 12 05:31:17 2009 (+485 ms)
        Time since last state change: 0 days, 00:08:52.130
        Effective State: DOWN
        Client Idle Timeout: 180 sec
        Down state flush: ENABLED
        Disable Primary Vserver On Down : DISABLED
        No. of Bound Services :  1 (Total)       1 (Active)
        Configured Method: LEASTCONNECTION
        Mode: IP
        Persistence: NONE
        Vserver IP and Port insertion: OFF
        Push: DISABLED  Push VServer:
        Push Multi Clients: NO
        Push Label Rule: none

1) ssl1 (10.102.29.252: 80) - HTTP State: UP    Weight: 1
 Done
```

To unbind a service from a virtual server by using the command line interface

At the command prompt, type the following command:

unbind lb vserver <name> <serviceName>

Example

```
unbind lb vserver vssl ssl1
```

To bind a service to a virtual server by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers.
2. Open a virtual server, and click in the Services section to bind a service to the virtual server.

Adding or Updating a Certificate-Key Pair

For any SSL transaction, the server needs a valid certificate and the corresponding private and public key pair. The SSL data is encrypted with the server's public key, which is available through the server's certificate. Decryption requires the corresponding private key.

Because the NetScaler appliance offloads SSL transactions from the server, the server's certificate and private key must be present on the appliance, and the certificate must be paired with its corresponding private key. This certificate-key pair must then be bound to the virtual server that processes the SSL transactions.
Note: From release 11.0, the default certificate on a NetScaler appliance is 2048-bits. In earlier builds, the default certificate was 512-bits or 1024-bits. After upgrading to release 11.0, you must delete all your old certificate-key pairs starting with "ns-", and then restart the appliance to automatically generate a 2048-bit default certificate.
Both the certificate and the key must be in local storage on the NetScaler appliance before they can be added to the appliance. If your certificate or key file is not on the appliance, upload it to the appliance before you create the pair.
Important: Certificates and keys are stored in the /nsconfig/ssl directory by default. If your certificates or keys are stored in any other location, you must provide the absolute path to the files on the NetScaler appliance. The NetScaler FIPS appliances do not support external keys (non-FIPS keys). On a FIPS appliance, you cannot load keys from a local storage device such as a hard disk or flash memory. The FIPS keys must be present in the Hardware Security Module (HSM) of the appliance.

On a NetScaler MPX appliance and a NetScaler FIPS appliance, only RSA private keys are supported. On a VPX virtual appliance, both RSA and DSA private keys are supported. On an SDX appliance if SSL chips are assigned to an instance, then only RSA private keys are supported. However, if SSL chips are not assigned to an instance, then both RSA and DSA private keys are supported. In all the cases, you can bind a CA certificate with either RSA or DSA keys.

Set the notification period and enable the expiry monitor to issue a prompt before the certificate expires.

The NetScaler appliance supports the following input formats of the certificate and the private-key files:

- PEM - Privacy Enhanced Mail
- DER - Distinguished Encoding Rule
- PFX - Personal Information Exchange

The format is automatically detected by the software. Therefore, you are no longer required to specify the format in the inform parameter. If you do specify the format (correct or incorrect), it is ignored by the software. The format of the certificate and the key file must be the same.
Note: A certificate must be signed by using one of the following hash algorithms:

- MD5
- SHA-1
- SHA-224
- SHA-256
- SHA-384 (supported only on the front end)
- SHA-512 (supported only on the front end)

An MPX appliance supports certificates of 512 or more bits, up to the following sizes:

- 4096-bit server certificate on the virtual server
- 4096-bit client certificate on the service
- 4096-bit CA certificate (includes intermediate and root certificates)
- 4096-bit certificate on the back-end server
- 4096-bit client certificate (if client authentication is enabled on the virtual server)

A VPX virtual appliance supports certificates of 512 or more bits, up to the following sizes:

- 4096-bit server certificate on the virtual server
- 4096-bit client certificate on the service
- 4096-bit CA certificate (includes intermediate and root certificates)
- 2048-bit certificate on the back-end server
- 2048-bit client certificate (if client authentication is enabled on the virtual server)

Note

A NetScaler SDX appliance supports certificates of 512 or more bits. Each NetScaler VPX instance hosted on the appliance supports the certificate sizes listed above for a VPX virtual appliance. However, if an SSL chip is assigned to an instance, that instance supports the certificate sizes supported by an MPX appliance.

To add a certificate-key pair by using the command line interface

At the command prompt, type the following commands to add a certificate-key pair and verify the configuration:

- add ssl certKey <certkeyName> -cert <string>[(-key <string> [-password]) | -fipsKey <string>] [-inform (DER | PEM)] [<passplain>] [-expiryMonitor (ENABLED | DISABLED) [-notificationPeriod <positive_integer>]]
- show ssl certKey [<certkeyName>]

Example

```
> add ssl certKey sslckey -cert server_cert.pem -key server_key.pem -password ssl -expiryMon
 Done
Note: For FIPS appliances, replace -key with -fipskey
> show ssl certKey sslckey
        Name: sslckey           Status: Valid,   Days to expiration:8418
        Version: 3
        Serial Number: 01
        Signature Algorithm: md5WithRSAEncryption
        Issuer:  C=US,ST=SJ,L=SJ,O=NS,OU=NSSSL,CN=www.root.com
        Validity
                Not Before: Jul 15 02:25:01 2005 GMT
                Not After : Nov 30 02:25:01 2032 GMT
        Subject:  C=US,ST=SJ,L=SJ,O=NS,OU=NSSSL,CN=www.server.com
        Public Key Algorithm: rsaEncryption
        Public Key size: 2048
 Done
```

To update or remove a certificate-key pair by using the command line interface

To modify the expiry monitor or notification period in a certificate-key pair, use the set ssl certkey command. To replace the certificate or key in a certificate-key pair, use the update ssl certkey command. The update ssl certkey command has an additional parameter for overriding the domain check. For both commands, enter the name of an existing certificate-key pair. To remove an SSL certificate-key pair, use the rm ssl certkey command, which accepts only the <certkeyName> argument.

To add or update a certificate-key pair by using the configuration utility

Navigate to Traffic Management > SSL > Certificates, and configure a certificate-key pair.

Binding the Certificate-Key Pair to the SSL-Based Virtual Server

An SSL certificate is an integral element of the SSL encryption and decryption process. The certificate is used during an SSL handshake to establish the identity of the SSL server.

The certificate being used for processing SSL transactions must be bound to the virtual server that receives the SSL data. If you have multiple virtual servers receiving SSL data, a valid certificate-key pair must be bound to each of them.

You can use a valid, existing SSL certificate that you have uploaded to the NetScaler appliance. As an alternative for testing purposes, you can create your own SSL certificate on the appliance. Intermediate certificates created by using a FIPS key on the NetScaler cannot be bound to an SSL virtual server.

As a part of the SSL handshake, in the certificate request message during client authentication, the server lists the distinguished names (DNs) of all the certificate authorities (CAs) bound to the server from which it will accept a client certificate. If you do not want the DN name of a specific CA certificate to be sent to the SSL client, set the `skipCA` flag. This setting indicates that the particular CA certificate's distinguished name should not be sent to the SSL client.

For details on how to create your own certificate, see Managing Certificates.

Note: Citrix recommends that you use only valid SSL certificates that have been issued by a trusted certificate authority.

To bind an SSL certificate-key pair to a virtual server by using the command line interface

At the command prompt, type the following commands to bind an SSL certificate-key pair to a virtual server and verify the configuration:

- bind ssl vserver <vServerName> -certkeyName <certificate-KeyPairName> -CA -skipCAName
- show ssl vserver <vServerName>

Example

```
 > bind ssl vs vs1 -certkeyName cert2 -CA -skipCAName
 Done
 > sh ssl vs vs1
 Advanced SSL configuration for VServer vs1:
 DH: DISABLED
 Ephemeral RSA: ENABLED Refresh Count: 0
 Session Reuse: ENABLED Timeout: 120 seconds
 Cipher Redirect: DISABLED
 SSLv2 Redirect: DISABLED
 ClearText Port: 0
 Client Auth: DISABLED
 SSL Redirect: DISABLED
 Non FIPS Ciphers: DISABLED
 SNI: DISABLED
 SSLv2: DISABLED SSLv3: ENABLED TLSv1: ENABLED
 Push Encryption Trigger: Always
 Send Close-Notify: YES
 1) CertKey Name: cert1 CA Certificate OCSPCheck: Optional CA_Name Sent
 2) CertKey Name: cert2 CA Certificate OCSPCheck: Optional CA_Name Skipped
 1) Cipher Name: DEFAULT
 Description: Predefined Cipher Alias
Done
```

To unbind an SSL certificate-key pair from a virtual server by using the command line interface

If you try to unbind a certificate-key pair from a virtual server by using the `unbind ssl certKey <certkeyName>` command, an error message appears because the syntax of the command has changed. At the command prompt, type the following command:

unbind ssl vserver <vServerName> -certkeyName <string>

Example

```
 unbind ssl vserver vssl -certkeyName sslckey
```

To bind an SSL certificate-key pair to a virtual server by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers.
2. Open an SSL virtual server and, in Advanced Settings, click SSL Certificate.
3. Bind a server certificate or CA certificate to the virtual server. To add a server certificate as an SNI certificate, select Server Certificate for SNI.

Configuring an SSL Virtual Server for Secure Hosting of Multiple Sites

Virtual hosting is used by Web servers to host more than one domain name with the same IP address. The NetScaler supports hosting of multiple secure domains by offloading SSL processing from the Web servers using transparent SSL services or virtual server-based SSL offloading. However, when multiple Web sites are hosted on the same virtual server, the SSL handshake is completed before the expected host name is sent to the virtual server. As a result, the NetScaler cannot determine which certificate to present to the client after a connection is established. This problem is resolved by enabling Server Name Indication (SNI) on the virtual server. SNI is a Transport Layer Security (TLS) extension used by the client to provide the host name during handshake initiation. The NetScaler appliance compares this host name to the common name and, if it does not match, compares it to the subject alternative name (SAN). If the name matches, the appliance presents the corresponding certificate to the client.

A wildcard SSL Certificate helps enable SSL encryption on multiple subdomains if the domains are controlled by the same organization and share the same second-level domain name. For example, a wildcard certificate issued to a sports network using the common name "*.sports.net" can be used to secure domains, such as "login.sports.net" and "help.sports.net" but not "login.ftp.sports.net."

Note: On a NetScaler appliance, only domain name, URL, and email ID DNS entries in the SAN field are compared.Â

You can bind multiple server certificates to a single SSL virtual server or transparent service using the -SNICert option. These certificates are issued by the virtual server or service if SNI is enabled on the virtual server or service. You can enable SNI at any time.

To bind multiple server certificates to a single SSL virtual server by using the command line interface

At the command prompt, type the following commands to configure SNI and verify the configuration:

- set ssl vserver <vServerName>@ [-SNIEnable (ENABLED | DISABLED)]
- bind ssl vserver <vServerName>@ -certkeyName <string> -SNICert
- show ssl vserver <vServerName>

  To bind multiple server certificates to a transparent service by using the NetScaler command line, replace vserver with service and vservername with servicename in the above commands.

  Note: The SSL service should be created with -clearTextPort 80 option.

Example

```
set ssl vserver v1 -snI ENABLED
bind ssl vserver v1 -certkeyName serverabc -SNICert
sh ssl vserver v1
Advanced SSL configuration for VServer v1:
â€¦
SSL Redirect: DISABLED
Non FIPS Ciphers: DISABLED
SNI: ENABLED
SSLv2: DISABLED SSLv3: ENABLED TLSv1: ENABLED
1)CertKey Name: servercert Server Certificate
1)CertKey Name: abccert Server Certificate for SNI
2)CertKey Name: xyzcert Server Certificate for SNI
3)CertKey Name: startcert Server Certificate for SNI
1)Cipher Name: DEFAULT
Description: Predefined Cipher Alias
Done
```

To bind multiple server certificates to a single SSL virtual server or transparent SSL service by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers.
2. Open an SSL virtual server and, in Certificates, click Server Certificate.
3. Add a new certificate or select a certificate from the list, and select Server Certificate for SNI.
4. In Advanced Settings, click SSL Parameters.
5. Select SNI Enable.

Support for SNI on the Back-End Service

The NetScaler appliance now supports Server Name Indication (SNI) at the back end. That is, the common name is sent as the server name in the client hello to the back-end server for successful completion of the handshake. In addition to helping meet federal system integrator customer security requirements, this enhancement provides the advantage of using only one port instead of opening hundreds of different IP addresses and ports on a firewall.

Federal system integrator customer security requirements include support for Active Directory Federation Services (ADFS) 3.0 in 2012R2 and WAP servers. This requires supporting SNI at the back end on a NetScaler appliance.

Note

For SNI to work, the server name in the client hello must match the host name configured on the back-end service that is bound to an SSL virtual server. For example, if the host name of the virtual server is https://www.mail.example.com, the SNI-enabled back-end service must be configured with the server name as https://www.mail.example.com, and this host name must match the server name in the client hello.

To configure SNI on the back-end service by using the NetScaler command line

At the command prompt, type:

add service <name> <IP> <serviceType> <port>

add lb vserver <name> <IPAddress> <serviceType> <port>

bind lb vserver <name> <serviceName>

set ssl service <serviceName> -**SNIEnable ENABLED** -**commonName** <string>

Example

```
add service service_ssl 10.217.193.2 SSL 443

add lb vserver ssl-vs 10.1.1.1 SSL 443

bind lb vserver ssl-vs service_ssl

set ssl service service_ssl -SNIEnable ENABLED â€“commonName www.example.com
```

To configure SNI on the back-end service by using the NetScaler GUI

1. Navigate to **Traffic Management** > **Load Balancing** > **Services**.
2. Select an SSL service, and in **Advanced Settings**, select **SSL Parameters**.
3. Select **SNI Enable**.

Binding a Secure Monitor to an SNI-Enabled Back-End Service

You can also bind secure monitors of type HTTP-ECV or TCP-ECV to the back-end services that support SNI. To do this, the custom header in the monitor must be set to the server name that is sent as the SNI extension in the client hello.

To configure and bind a secure monitor to an SNI-enabled back-end service by using the NetScaler command line

At the command prompt, type:

add lb monitor <monitorName> <type>

set lb monitor <monitorName> <type> -**customHeaders** <string>

bind service <name> -**monitorName** <string>

Example

```
> add monitor https-ecv-mon http-ecv

 Done

> set monitor https-ecv-mon HTTP-ECV -customHeaders "Host: example.com\r\n"

 Done

> bind service ssl_service â€“monitorName https-ecv
```

To configure and bind a secure monitor to an SNI-enabled back-end service by using the NetScaler GUI

1. Navigate to **Traffic Management** > **Load Balancing** > **Monitor**.
2. Add a monitor of type **HTTP-ECV** or **TCP-ECV**, and specify a **Custom Header**.
3. Click **Create**.
4. Navigate to **Traffic Management** > **Load Balancing** > **Services**.
5. Select an SSL service and click **Edit**.
6. In **Monitors**, click **Add Binding**, select the monitor created in step 3, and click **Bind**.

Configuring a DTLS Virtual Server

The SSL and TLS protocols have traditionally been used to secure streaming traffic. Both of these protocols are based on TCP, which is very slow. In addition, TLS cannot handle lost or reordered packets.

UDP is the preferred protocol for audio and video applications, such as Lync, Skype, iTunes, YouTube, training videos, and flash. However, UDP is not secure or reliable. The DTLS protocol is designed to secure data over UDP and is used for applications such as media streaming, VOIP, and online gaming for communication. In DTLS, each handshake message is assigned a specific sequence number within that handshake. When a peer receives a handshake message, it can quickly determine whether that message is the next one expected. If it is, the peer processes the message. If not, the message is queued for handling after all the previous messages have been received.

You must create a DTLS virtual server and a service of type UDP. By default, a DTLS profile (`nsdtls_default_profile`) is bound to the virtual server. Optionally, you can create and bind a user-defined DTLS profile to the virtual server.

Note: RC4 ciphers are not supported on a DTLS virtual server.

To create a DTLS configuration by using the command line

At the command prompt, type:

```
        add lb vserver <vserver_name> DTLS <IPAddress>  <port>
add service  <service_name>  <IPAddress> UDP 443
bind lb vserver  <vserver_name>  <udp_service_name>
```

The following steps are optional:

```
add dtlsProfile dtls1 -maxretryTime <positive_integer>
set ssl vserver <vserver_name> -dtlsProfileName <dtls_profile_name>
```

To create a DTLS configuration by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers.
2. Create a virtual server of type DTLS, and bind a UDP service to the virtual server.
3. A default DTLS profile is bound to the DTLS virtual server. To bind a different profile, in SSL Parameters, select a different DTLS profile. To create a new profile, click the plus (+) next to DTLS Profile.

Example

The following example is for an end-to-end DTLS configuration:

```
> enable ns feature SSL LB
> add server s1 10.102.59.190
> add service svc1 s1 UDP 32000
> add lb vserver lb1 DTLS 10.102.59.244 443
> add ssl certKey servercert -cert server_cert.pem -key server_key.pem
> bind ssl vserver lb1 -certkeyname servercert
> bind lb vserver lb1 svc1

> sh lb vserver lb1
        lb1 (10.102.59.244:443) - DTLS  Type: ADDRESS
        State: UP
        Last state change was at Tue May 20 16:41:27 2014
        Time since last state change: 0 days, 00:01:39.120
        Effective State: UP
        Client Idle Timeout: 120 sec
        Down state flush: ENABLED
        Disable Primary Vserver On Down : DISABLED
        Appflow logging: ENABLED
        No. of Bound Services :  1 (Total)       1 (Active)
        Configured Method: LEASTCONNECTION
        Current Method: Round Robin, Reason: A new service is bound
        Mode: IP
        Persistence: NONE
        L2Conn: OFF
        Skip Persistency: None
        IcmpResponse: PASSIVE
        RHIstate: PASSIVE
        New Service Startup Request Rate: 0 PER_SECOND, Increment Interval: 0
        TD: 0
        Mac mode Retain Vlan: DISABLED
        DBS_LB: DISABLED
        Process Local: DISABLED

        1 bound service:
1) svc1 (10.102.59.190: 32000) - UDP State: UP  Weight: 1
 Done
>
> sh ssl vserver lb1

        Advanced SSL configuration for VServer lb1:
        DH: DISABLED
        Ephemeral RSA: ENABLED          Refresh Count: 0
        Session Reuse: ENABLED          Timeout: 1800 seconds
        Cipher Redirect: DISABLED

        ClearText Port: 0
        Client Auth: DISABLED
        SSL Redirect: DISABLED
        Non FIPS Ciphers: DISABLED
        SNI: DISABLED
        DTLSv1: ENABLED
        Send Close-Notify: YES

        DTLS profile name: nsdtls_default_profile

        1 bound certificate:

1)      CertKey Name: servercert        Server Certificate

        1 configured cipher:

1)      Cipher Name: DEFAULT
        Description: Predefined Cipher Alias
 Done

> sh dtlsProfile nsdtls_default_profile
1) Name: nsdtls_default_profile
 PMTU Discovery: DISABLED
 Max Record Size: 1460 bytes
 Max Retry Time: 3 sec
 Hello Verify Request: DISABLED
 Terminate Session: DISABLED
 Max Packet Count: 120 bytes
 Done
```

Features not supported by a DTLS virtual server

The following options cannot be enabled on a DTLS virtual server:

- SSLv2
- SSLv3
- TLSv1
- TLSv1.1
- TLSv1.2
- Push encrypt trigger
- SSLv2Redirect
- SSLv2URL
- SNI
- Secure renegotiation

Parameters not used by a DTLS virtual server

The following SSL parameters, even if set, are ignored by a DTLS virtual server:

- Encryption trigger packet count
- PUSH encryption trigger timeout
- SSL quantum size
- Encryption trigger timeout
- Subject/Issuer Name Insertion Format

DTLS Profile

A DTLS profile with the default settings is automatically bound to a DTLS virtual server. However, you can create a new DTLS profile with specific settings to suit your requirement.

To create a DTLS profile by using the command line

- add ssl dtlsProfile <name>
- show ssl dtlsProfile<name>

Example

```
> add dtlsProfile dtls1 -helloVerifyRequest ENABLED -maxretryTime 4
 Done
> show dtlsProfile dtls1
1)      Name: dtls1
        PMTU Discovery: DISABLED
        Max Record Size: 1460 bytes
        Max Retry Time: 4 sec
        Hello Verify Request: ENABLED
        Terminate Session: DISABLED
        Max Packet Count: 120 bytes

 Done
```

To create a DTLS profile by using the configuration utility

Navigate to System > Profiles > DTLS Profiles and configure a new profile.

Importing SSL Files from Remote Hosts

You can now import SSL resources, such as certificates, private keys , CRLs, and DH keys, from remote hosts even if FTP access to these hosts is not available. This is especially helpful in environments where shell access to the remote host is restricted. Default folders are created in /nsconfig/ssl as follows:

- For certificate files: /nsconfig/ssl/certfile
- For private keys: the /nsconfig/ssl/keyfile
- For CRLs: /var/netscaler/ssl/crlfile
- For DH keys: /nsconfig/ssl/dhfile

Imports from both HTTP and HTTPS servers are supported. However, the import fails if the file is on an HTTPS server that requires client certificate authentication for access.

Note: The import command is not stored in the configuration (ns.conf) file, because reimporting the file after a restart might cause an error.

To import a certificate file from a remote host by using the command line

At the command prompt, type:

```
import ssl certFile [<name>] [<src>]
```

Example

```
import ssl certfile my-certfile http://www.example.com/file_1
> show ssl certfile
     Name : my-certfile
     URL : http://www.example.com/file_1
```

To remove a certificate file, use the rm ssl certFile command, which accepts only the <name> argument.

To import a key file from a remote host by using the command line

At the command prompt, type:

```
import ssl keyFile [<name>] [<src>]
```

Example

```
import ssl keyfile my-keyfile http://www.example.com/key_file
> show ssl keyfile
     Name : my-keyfile
     URL : http://www.example.com/key_file
```

To remove a key file, use the rm ssl keyFile command, which accepts only the <name> argument.

To import a CRL file from a remote host by using the command line

At the command prompt, type:

```
import ssl crlFile [<name>] [<src>]
```

Example

```
import ssl crlfile my-crlfile http://www.example.com/crl_file
> show ssl crlfile
     Name : my-crlfile
     URL : http://www.example.com/crl_file
```

To remove a CRL file, use the rm ssl crlFile command, which accepts only the <name> argument.

To import a DH file from a remote host by using the command line

At the command prompt, type:

```
import ssl dhFile [<name>] [<src>]
```

Example

```
import ssl dhfile my-dhfile http://www.example.com/dh_file
> show ssl dhfile
     Name : my-dhfile
     URL : http://www.example.com/dh_file
```

To remove a DH file, use the rm ssl dhFile command, which accepts only the <name> argument.

To import an SSL resource by using the configuration utility

Navigate to Traffic Management > SSL > Imports, and then select the appropriate tab.

Enabling Stricter Control on Client Certificate Validation

The NetScaler appliance accepts valid Intermediate-CA certificates if they are issued by a single Root-CA. That is, if only the Root-CA certificate is bound to the virtual server, and any intermediate certificate sent with the client certificate is validated by that Root-CA, the appliance trusts the certificate chain and the handshake is successful.

However, if a client sends a chain of certificates in the handshake, none of the intermediate certificates can be validated by using a CRL or OCSP responder unless that certificate is bound to the SSL virtual server. Therefore, even if one of the intermediate certificates is revoked, the handshake is successful. As part of the handshake, the SSL virtual server sends the list of CA certificates that are bound to it. For stricter control, you can configure the SSL virtual server to accept only a certificate that is signed by one of the CA certificates bound to that virtual server. To do so, you must enable the `ClientAuthUseBoundCAChain` setting in the SSL profile bound to the virtual server. The handshake fails if the client certificate is not signed by one of the CA certificates bound to the virtual server.

For example, say two client certificates, clientcert1 and clientcert2, are signed by the intermediate certificates Int-CA-A and Int-CA-B, respectively. The intermediate certificates are signed by the root certificate Root-CA. Int-CA-A and Root-CA are bound to the SSL virtual server. In the default case (`ClientAuthUseBoundCAChain` disabled), both clientcert1 and clientcert2 are accepted. However, if `ClientAuthUseBoundCAChain` is enabled, only clientcert1 is accepted by the NetScaler appliance

To enable stricter control on client certificate validation by using the command line

At the NetScaler command prompt, type:

- `set ssl profile <name> -ClientAuthUseBoundCAChain Enabled`
- `set ssl vserver <vServerName> -sslProfile <string>`

To enable stricter control on client certificate validation by using the configuration utility

1. Navigate to System > Profiles, select the SSL Profiles tab, and create an SSL profile, or select an existing profile.
2. Select Enable Client Authentication using bound CA Chain.
3. Navigate to Traffic Management > Load Balancing > Virtual Servers, and select an SSL virtual server.
4. In Advanced Settings, select SSL Profiles, and select the profile on which you enabled Enable Client Authentication using bound CA Chain.
5. Click OK, and then click Done.

Graceful Cleanup of SSL Sessions

Some operations, such as updating a certificate to replace a potentially exposed certificate, using a stronger key (2048-bit instead of 1024-bit), adding or removing a certificate to or from a certificate chain, or changing any of the SSL parameters, should clean the SSL sessions gracefully instead of abruptly terminating the sessions.

From build 64.x, existing SSL connections do not break if you update the SSL certificate, cipher list, or SSL parameters. That is, all existing connections continue using the current settings until the sessions are closed, but all new connections use the new certificate or settings. To clear the sessions immediately after a configuration change, you must disable and reenable each entity.

Important: Connections that are in the middle of a handshake, or sessions that are renegotiating, are terminated. Session reuse is not allowed. Additionally, session multiplexing reuse at the back end is not allowed.

If you change a front-end parameter, such as on an SSL virtual server, only the front end connections are affected. If you change a back-end parameter, such as a parameter on an SSL service or service group, only the back-end connections are affected. Changes such as ciphers and certificates apply to both front-end and back-end connections.

The following configuration commands or changes trigger a graceful session cleanup on all affected SSL entities:

1. set ssl vserver command
2. set ssl service command
3. set ssl servicegroup command
4. set ssl profile command
5. set ssl cipher <cipherGroupName> command
6. Binding, unbinding, and reordering ciphers
7. Binding and unbinding ecccurves
8. Inserting, removing, linking and unlinking a certificate

Enhanced SSL Profiles Infrastructure Overview

Vulnerabilities in SSLv3 and RC4 implementation have emphasized the need to use the latest ciphers and protocols to negotiate the security settings for a network connection. Implementing any changes to the configuration, such as disabling SSLv3 across thousands of SSL end points, is a cumbersome process. Therefore, settings that were part of the SSL end points configuration have been moved to the SSL profiles, along with the default ciphers. To implement changes in the configuration, including cipher support, you need only modify the profile that is bound to the entities.

The default SSL profiles (default front-end and default back-end) contain all the default ciphers and ECC curves, in addition to the settings that were part of the old profiles. Sample outputs for the default profiles are provided in the appendix. The Enable Default Profile operation automatically binds the default front-end profile to all front-end entities, and the default back-end profile to all back-end entities. You can modify a default profile to suit your deployment. You can also create custom profiles and bind them to SSL entities.

Important

After the upgrade, if you enable the default profiles, you cannot undo the changes. That is, the profiles cannot be disabled. Save the configuration and create a copy of the configuration file (ns.conf) before enabling the profiles.

By default, some SSL parameters, called *global parameters*, apply to all the SSL end points. However, if a profile is bound to an SSL end point, the global parameters do not apply. The settings specified in the profile apply instead.

Points to Note

1. A profile can be bound to multiple virtual servers, but a virtual server can have only one profile bound to it.
2. You cannot delete a profile that is bound to a virtual server without first unbinding the profile.
3. A cipher or cipher group can be bound to multiple profiles at different priorities.
4. A profile can have multiple ciphers and cipher groups bound at different priorities.
5. Changes to a cipher group are immediately reflected in all the profiles and in all the virtual servers that one of the profiles is bound to.
6. If a cipher suite is part of a cipher group, you cannot remove the cipher suite from the profile without first editing the cipher group to remove the specific cipher suite.
7. If you do not assign a priority to a cipher suite or cipher group that you attach to a profile, it is assigned the lowest priority within the profile.
8. You can create a custom cipher group (also called a user-defined cipher group) from existing cipher groups and ciphe suites. If you create cipher group A and add existing cipher groups X and Y to it, in that order, cipher group Y is assigned at a lower priority than cipher group X. That is, the group that is added first has a higher priority.
9. If a cipher suite is already part of a cipher group that is attached to a profile, and the same cipher suite is part of another cipher group that is also attached to the same profile, the cipher suite is not added again as part of the seconc cipher group. The cipher suite at the higher priority is in effect when traffic is processed.
10. Cipher groups are not expanded in the profile. As a result, the number of lines in the configuration file (ns.conf) is greatly reduced. For example, if there are a thousand SSL virtual servers to which two cipher groups are bound, and each cipher group contains 15 ciphers, expansion would result in 30*1000 entries related to ciphers in the configuration file. With the new profile, it would have only two entries: one for each cipher group that is bound to a profile.
11. Creating a user defined cipher group from existing ciphers and cipher groups is a copy-paste operation. Any changes in the original group are not reflected in the new group.
12. A user-defined cipher group lists all the profiles that it is a part of.
13. A profile lists all the SSL virtual server, services, and service groups that it is bound to.
14. If the default SSL profile feature is enabled, you must use the profile to set or change any of the attributes of a virtual server, service, service group, or an internal service.

Limitations

SSL profiles are not supported in a cluster setup, or with Admin Partitions.

Differences between the Old and the New SSL Profile Infrastructure

	Old Profile	**New Profile**
Ciphers and ECC Curves included in the profile	No	Yes
Inserting a cipher or cipher group in the middle of an existing list	Unbind all the ciphers and bind again in the order of the required priority.	Add a cipher and assign it a priority. If a priority is not specified, the cipher is assigned the lowest priority in the list.
Unbinding all the ciphers	> unbind ssl vserver <name> ciphername â€“ALL	unbind ssl profile â€“cipherName FlushAllCiphers (Release 11.0 build 64.x or later includes the FlushAllCiphers parameter for unbinding all the ciphers or cipher groups from a profile, because ALL is treated like a cipher group.)
State of SSLv3	n/a	Disabled on the default front-end profile (ns_default_ssl_profile_frontend). Note: Before you enable this profile, SSLv3 is enabled globally. After enabling the profile, SSLv3 is disabled on the front-end default profile.

Enabling the Default Profiles

Important

Save your configuration before you upgrade the software and enable the default profiles.

Upgrade the software to a build that supports the enhanced profile infrastructure, and then enable the default profiles. You can take one of two approaches depending on your specific deployment. If your deployment has a common SSL configuration across end points, see Use Case 1. If your deployment has a large SSL configuration and the SSL parameters and ciphers are not common among end points, see Use Case 2.

Upgrade the software to a build that supports the enhanced profile infrastructure, and then enable the default profiles. You can take one of two approaches depending on your specific deployment. If your deployment has a common SSL configuration across end points, see Use Case 1. If your deployment has a large SSL configuration and the SSL parameters and ciphers are not common among end points, see Use Case 2.

Note

A single operation (Enable Default Profile or set ssl parameter -defaultProfile ENABLED) enables (binds) both the default front-end profile and the default back-end profile.

To save the configuration by using the NetScaler command line

At the command prompt, type:

> **save config**

> **shell**

root@ns# **cd /nsconfig**

root@ns# **cp ns.conf** ns.conf.NS<currentreleasenumber><currentbuildnumber>

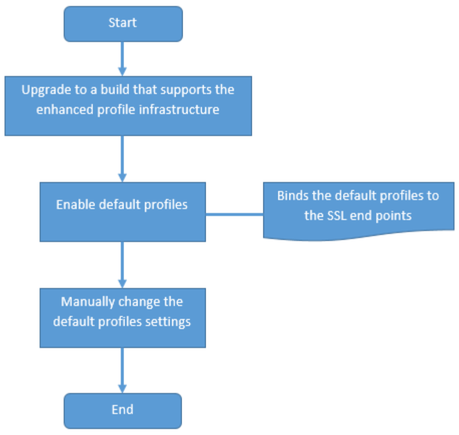
Example

```
> save config

> shell

root@ns# cd /nsconfig

root@ns# cp ns.conf ns.conf.NS.11.0.jun.16
```

Use Case 1

After you enable the default profiles, they are bound to all the SSL end points. The default profiles are editable. If your deployment uses most of the default settings and changes only a few parameters, you can edit the default profiles. The changes are immediately reflected across all the end points.

The following flowchart explains the steps that you must perform:


Start
Upgrade to a build that supports the enhanced profile infrastructure
Enable default profiles
Binds the default profiles to the SSL end points
Manually change the default profiles settings
End

1. For information about upgrading the software, see Upgrading the System Software.

2. Enable the default profiles by using the NetScaler command line or GUI.

- At the command line, type: **set ssl parameter -defaultProfile ENABLED**
- If you prefer to use the GUI, navigate to **Traffic Management** > **SSL** > **Change advanced SSL settings**, scroll down, and select **Enable Default Profile**.

If a profile was not bound to an end point before the upgrade, a default profile is bound to the SSL end point. If a profile was bound to an end point before the upgrade, the same profile is bound after the upgrade, and default ciphers are added to the profile.

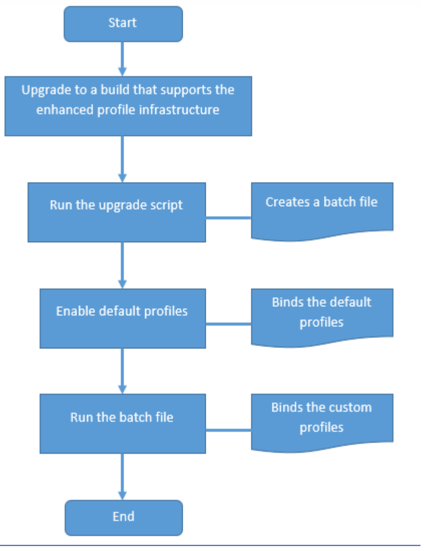
3. (Optional) Manually change any settings in the default profile.

- At the command line, type: **set ssl profile** <name> followed by the parameters to modify.
- If you prefer to use the GUI, navigate to **System** > **Profiles**. In **SSL Profiles**, select a profile and click **Edit**.

Use Case 2

If your deployment uses specific settings for most of the SSL entities, you can run a script that automatically creates custom profiles for each end point and binds them to the end point. Use the procedure detailed in this section to retain the SSL settings for all the SSL end points in your deployment. After upgrading the software, download and run a migration script to capture the SSL-specific changes. The output of running this script is a batch file. Enable the default profiles and then apply the commands in the batch file. See the appendix for a sample migration of the SSL configuration after upgrade.

The following flowchart explains the steps that you must perform:


Start
Upgrade to a build that supports the enhanced profile infrastructure
Run the upgrade script
Creates a batch file
Enable default profiles
Binds the default profiles
Run the batch file
Binds the custom profiles
End

1. For information about upgrading the software, see Upgrading the System Software.

2. Download and run a script to capture the SSL-specific changes. In addition to other migration activities, the script analyzes the old ns.conf file and moves any special settings (settings other than the default) from an SSL end point configuration to a custom profile. You must enable the default profiles after the upgrade for the configuration changes to apply.

To download the script, log on to https://www.citrix.com/. On the **Downloads** tab, select **NetScaler ADC**, and then select the release (for example, Release 11.0). Within the release, in **Firmware**, select a build (for example, 64.34). The SSL Default Profile Script is available in **Additional Components**.

Note

When running the migration script, you can choose to automatically generate the profile names, or you can prompt the user for the profile names interactively. The migration script checks the following and creates profiles accordingly.

- End points with the default settings and similar ciphers and cipher group settings: The script creates one profile.
- End points with the default settings and with different cipher groups or different priorities for the ciphers/cipher groups: In each case, the script creates a user-defined cipher group, binds it to a profile, and binds each profile to the appropriate end points.
- End points with the default settings and default ciphers: A default profile is bound to the end point.

To run the script, at the command prompt, type:

./default_profile_script /nsconfig/ns.conf -b > <output file name>

Note

You must run this command from the folder in which you store the script.

3. Enable the default profiles by using the NetScaler command line or GUI.

- At the command line, type: **set ssl parameter -defaultProfile ENABLED**
- If you prefer to use the GUI, navigate to **Traffic Management** > **SSL** > **Change advanced SSL settings**, scroll down, and select **Enable Default Profile**.

If a profile was not bound to an end point before the upgrade, a default profile is bound to the SSL end point. If a profile was bound to an end point before the upgrade, the same profile is bound after the upgrade, and default ciphers are added to the profile.

4. Apply the commands in the text file (output of running the migration script) to the configuration. After you apply the commands in the text file, custom profiles are created for end points for which default parameters and ciphers have been changed, and the custom profiles are automatically bound to the end points.

Loading an Old Configuration

Enabling the default profiles is not reversible. However, if you decide that your deployment does not require the default profiles, you can load an older configuration that you saved before you enabled the default profiles. The changes are effective after you restart the appliance.

To load an old configuration by using the NetScaler command line

At the command prompt, type:

```
> shell

root@ns# clear config

root@ns# cd /nsconfig

root@ns# cp ns.conf.NS.11.0.jun.16 ns.conf

root@ns# reboot
```

Appendix A: Sample Migration of the SSL Configuration after Upgrade

Sample settings on an SSL virtual server, service, and service group are shown below. On the virtual server, client authentication is ENABLED (default is DISABLED), and the AES cipher group is bound to the virtual server. On the service, server authentication is ENABLED (default is DISABLED), and the AES cipher group is bound to the service. The service group has the default settings.

```
>  sh ssl vserver v1


    Advanced SSL configuration for VServer v1:

    DH: DISABLED

    Ephemeral RSA: ENABLED        Refresh Count: 0

    Session Reuse: ENABLED        Timeout: 120 seconds

    Cipher Redirect: DISABLED

    SSLv2 Redirect: DISABLED

    ClearText Port: 0

    Client Auth: ENABLED Client Cert Required: Mandatory

    SSL Redirect: DISABLED

    Non FIPS Ciphers: DISABLED

    SNI: DISABLED

    SSLv2: DISABLED SSLv3: ENABLED  TLSv1.0: ENABLED  TLSv1.1: DISABLED  TLSv1.2: DISABLED

    Push Encryption Trigger: Always

    Send Close-Notify: YES


    ECC Curve: P_256, P_384, P_224, P_521


1)   CertKey Name: mycertkey    Server Certificate




1)   Cipher Name: AES

    Description: Predefined Cipher Alias

 Done


> sh ssl service svc1


    Advanced SSL configuration for Back-end SSL Service svc1:

    DH: DISABLED

    Ephemeral RSA: DISABLED

    Session Reuse: ENABLED        Timeout: 300 seconds
    Cipher Redirect: DISABLED

    SSLv2 Redirect: DISABLED

    ClearText Port: 0

    Server Auth: ENABLED

    SSL Redirect: DISABLED

    Non FIPS Ciphers: DISABLED

    SNI: DISABLED

    SSLv2: DISABLED SSLv3: ENABLED  TLSv1.0: ENABLED  TLSv1.1: DISABLED  TLSv1.2: DISABLED

    Send Close-Notify: YES



1)   Cipher Name: AES

    Description: Predefined Cipher Alias

 Done
```

```
> sh ssl serviceGroup

1) Service Group Name: sg1

    Session Reuse: ENABLED          Timeout: 300 seconds

    Server Auth: DISABLED

    Non FIPS Ciphers: DISABLED

    SSLv3: ENABLED  TLSv1.0: ENABLED

    Send Close-Notify: YES

 Done
```

The following procedure migrates the above configuration.

1. Save your configuration.

2. Run the migration script. You can redirect the output to a text file if you use the default names for the profiles. Type:

./default_profile_script /nsconfig/ns.conf -b > ssl_config.txt

Use an editor, such as vi, to view the changes. The output cannot be redirected if you provide the profile names interactively. The output is displayed on the console and you must copy and paste it into a text file to apply it to your configuration after the upgrade.

3. After the upgrade, enable the profile.

- At the command line, type: **set ssl parameter -defaultProfile ENABLED**
- In the GUI, navigate to **Traffic Management** > **SSL** > **Change advanced SSL settings**, scroll down and select **Enable Default Profile**.

The interim output for the three new profiles that are created for the virtual server, service, and service group, respectively, is shown below. The default profiles are bound to the end points until you apply the changes in the text file that was created after running the migration script.

```
> sh ssl vserver v1
```

```
Advanced SSL configuration for VServer v1:

Profile Name :ns_default_ssl_profile_frontend

1) CertKey Name: mycertkey Server Certificate

 Done

>

> sh ssl service svc1


Advanced SSL configuration for Back-end SSL Service svc1:

Profile Name :ns_default_ssl_profile_backend

 Done

>

> sh ssl serviceGroup sg1


Advanced SSL configuration for Back-end SSL Service Group sg1:

Profile Name :ns_default_ssl_profile_backend

 Done
```

4. You must now apply the configuration in ssl_config.txt to the current configuration, so that your non-default settings are applied after the upgrade.

batch -f /<path to the batch file>/ssl_config.txt

5. After applying the configuration, the output changes as follows:

```
> show ssl vserver v1


    Advanced SSL configuration for VServer v1:

    Profile Name :profile-002




1)   CertKey Name: mycertkey    Server Certificate

 Done


> show ssl service svc1


    Advanced SSL configuration for Back-end SSL Service svc1:

    Profile Name :profile-001

 Done
```

```
> show ssl serviceGroup sg1


    Advanced SSL configuration for Back-end SSL Service Group sg1:
    Profile Name :profile-003
 Done


> show ssl profile profile-002
1)   Name: profile-002    (Front-End)
    SSLv3: ENABLED  TLSv1.0: ENABLED  TLSv1.1: DISABLED  TLSv1.2: DISABLED
    Client Auth: ENABLED Client Cert Required: Mandatory
    Use only bound CA certificates: DISABLED
    Strict CA checks:          NO
    Session Reuse: ENABLED          Timeout: 120 seconds
    DH: DISABLED
    Ephemeral RSA: ENABLED          Refresh Count: 0
    Deny SSL Renegotiation          ALL
    Non FIPS Ciphers: DISABLED
    Cipher Redirect: DISABLED
    SSL Redirect: DISABLED
    Send Close-Notify: YES
    Push Encryption Trigger: Always
    PUSH encryption trigger timeout:     1 ms
    SNI: DISABLED
    Strict Host Header check for SNI enabled SSL sessions:          NO
    Push flag: 0x0 (Auto)
    SSL quantum size:          8 kB
    Encryption trigger timeout 100 mS
    Encryption trigger packet count:     45
    Subject/Issuer Name Insertion Format: Unicode



    ECC Curve: P_256, P_384, P_224, P_521


1)   Cipher Name: AES     Priority :1
    Description: Predefined Cipher Alias
```

```
1)   Vserver Name: v1
 Done



> show ssl profile profile-001
1)   Name: profile-001    (Back-End)
     SSLv3: ENABLED  TLSv1.0: ENABLED  TLSv1.1: DISABLED  TLSv1.2: DISABLED
     Server Auth: ENABLED
     Use only bound CA certificates: DISABLED
     Strict CA checks:          NO
     Session Reuse: ENABLED          Timeout: 120 seconds
     Deny SSL Renegotiation          ALL
     Non FIPS Ciphers: DISABLED
     Send Close-Notify: YES
     Push Encryption Trigger: Always
     PUSH encryption trigger timeout:     1 ms
     Push flag: 0x0 (Auto)
     SSL quantum size:          8 kB
     Encryption trigger timeout 100 mS
     Encryption trigger packet count:     45



     ECC Curve: P_256, P_384, P_224, P_521

1)   Cipher Name: AES     Priority :1
     Description: Predefined Cipher Alias

1)   Service Name: svc1
 Done

> show ssl profile profile-003
1)   Name: profile-003    (Back-End)
     SSLv3: ENABLED  TLSv1.0: ENABLED  TLSv1.1: DISABLED  TLSv1.2: DISABLED
     Server Auth: DISABLED
     Use only bound CA certificates: DISABLED
     Strict CA checks:          NO
```

```
    Session Reuse: ENABLED         Timeout: 120 seconds

    Deny SSL Renegotiation         ALL

    Non FIPS Ciphers: DISABLED

    Send Close-Notify: YES

    Push Encryption Trigger: Always

    PUSH encryption trigger timeout:     1 ms

    Push flag: 0x0 (Auto)

    SSL quantum size:          8 kB

    Encryption trigger timeout 100 mS

    Encryption trigger packet count:     45



    ECC Curve: P_256, P_384, P_224, P_521


1)  Cipher Name: ALL     Priority :1

    Description: Predefined Cipher Alias


1)  Service Name: sg1

 Done
```

Appendix B: Default Front-End and Back-End SSL Profile Settings

A default front-end profile has the following settings:

```
> sh ssl profile ns_default_ssl_profile_frontend

1)Name: ns_default_ssl_profile_frontend

    Configuration for Front-End SSL profile

    DH: DISABLED

    Ephemeral RSA: ENABLED         Refresh Count: 0

    Session Reuse: ENABLED         Timeout: 120 seconds

    Non FIPS Ciphers: DISABLED

    Cipher Redirect: ENABLED   Redirect URL: http://10.102.28.212/redirect.html

    Client Auth: DISABLED

    SSL Redirect: DISABLED

    SNI: DISABLED

    SSLv3: DISABLED TLSv1.0: ENABLED  TLSv1.1: ENABLED  TLSv1.2: ENABLED

    Push Encryption Trigger: Always

    PUSH encryption trigger timeout:     1 ms

    Send Close-Notify: YES

    Push flag: 0x0 (Auto)

    Deny SSL Renegotiation         NO

    SSL quantum size:         8 kB

    Strict CA checks:         NO

    Encryption trigger timeout 100 mS

    Encryption trigger packet count:     45

    Use only bound CA certificates: DISABLED

    Subject/Issuer Name Insertion Format: Unicode

    Strict Host Header check for SNI enabled SSL sessions:         NO


    ECC Curve: P_256, P_384, P_521


1)   Cipher Name: AES     Priority :2

    Description: Predefined Cipher Alias


1)   Vserver Name: v1  >>>>>>>>>>

2)   Vserver Name: nshttps-::1l-443 >>>>>>>>>>

3)   Vserver Name: nsrpcs-::1l-3008
```

```
4)   Vserver Name: nskrpcs-127.0.0.1-3009

5)   Vserver Name: nshttps-127.0.0.1-443

6)   Vserver Name: nsrpcs-127.0.0.1-3008

Done
```

A default back-end profile has the following settings:

```
> sh ssl profile ns_default_ssl_profile_backend

1)Name: ns_default_ssl_profile_backend

    Configuration for Back-End SSL profile

    Session Reuse: ENABLED          Timeout: 300 seconds

    Non FIPS Ciphers: DISABLED

    Server Auth: DISABLED

    SSLv3: DISABLED TLSv1.0: ENABLED  TLSv1.1: DISABLED  TLSv1.2: DISABLED

    Push Encryption Trigger: Always

    PUSH encryption trigger timeout:     1 ms

    Send Close-Notify: YES

    Push flag: 0x0 (Auto)

    Deny SSL Renegotiation          ALL

    SSL quantum size:          8 kB

    Strict CA checks:          NO

    Encryption trigger timeout 100 mS

    Encryption trigger packet count:     45

    Use only bound CA certificates: DISABLED


    ECC Curve: P_256, P_224, P_521


1)   Cipher Name: AES     Priority :1

    Description: Predefined Cipher Alias


2)   Cipher Name: RC4     Priority :2

    Description: Predefined Cipher Alias


1)   Service Name: s2 >>>>>>>>>>>>

2)   Service Name: s1 >>>>>>>>>>>>

Done
```

Managing Certificates

An SSL certificate, which is an integral part of any SSL transaction, is a digital data form (X509) that identifies a company (domain) or an individual. The certificate has a public key component that is visible to any client that wants to initiate a secure transaction with the server. The corresponding private key, which resides securely on the NetScaler appliance, is used to complete asymmetric key (or public key) encryption and decryption.

You can obtain an SSL certificate and key in either of the following ways:

- From an authorized certificate authority (CA), such as VeriSign
- By generating a new SSL certificate and key on the NetScaler appliance

Alternately, you can use an existing SSL certificate on the appliance.

Caution: Citrix recommends that you use certificates obtained from authorized CAs, such as VeriSign, for all your SSL transactions. Certificates generated on the NetScaler appliance should be used for testing purposes only, not in any live deployment.

To manage certificates, see the following sections:

- Obtaining a Certificate from a Certificate Authority
- Importing Existing Certificates and Keys
- Generating a Self-Signed Certificate
- Adding a Group of SSL Certificates
- Displaying a Certificate Chain
- Generating a Server Test Certificate
- Achieving Perfect Forward Secrecy With DHE
- Modifying and Monitoring Certificates and Keys
- Using Global Site Certificates
- Converting the Format of SSL Certificates for Import or Export

Obtaining a Certificate from a Certificate Authority

A certificate authority (CA) is an entity that issues digital certificates for use in public key cryptography. Certificates issued or signed by a CA are automatically trusted by applications, such as web browsers, that conduct SSL transactions. These applications maintain a list of the CAs that they trust. If the certificate being used for the secure transaction is signed by any of the trusted CAs, the application proceeds with the transaction.

To obtain an SSL certificate from an authorized CA, you must create a private key, use that key to create a certificate signing request (CSR), and submit the CSR to the CA. The only special characters allowed in the file names are underscore and dot.

Creating a Private Key

The private key is the most important part of a digital certificate. By definition, this key is not to be shared with anyone and should be kept securely on the NetScaler appliance. Any data encrypted with the public key can be decrypted only by using the private key.

The appliance supports two encryption algorithms, RSA and DSA, for creating private keys. You can submit either type of private key to the CA. The certificate that you receive from the CA is valid only with the private key that was used to create the CSR, and the key is required for adding the certificate to the NetScaler.

Caution: Be sure to limit access to your private key. Anyone who has access to your private key can decrypt your SSL data. All SSL certificates and keys are stored in the /nsconfig/ssl folder on the appliance. For added security, you can use the Data Encryption Standard (DES) or triple DES (3DES) algorithm to encrypt the private key stored on the appliance.
Note: The length of the SSL key name allowed includes the length of the absolute path name if the path is included in the key name.

To create an RSA private key by using the command line interface

At the command prompt, type the following command:

create ssl rsakey <keyFile> <bits> [-exponent (3 | F4)] [-keyform (DER | PEM)]

Example

```
> create ssl rsakey Key-RSA-1 2048 -exponent F4 -keyform PEM
```

To create a DSA private key by using the command line interface

At the command prompt, type the following command:

create ssl dsakey <keyfile> <bits> [-keyform (DER | PEM)]

Example

```
> create ssl dsakey Key-DSA-1 2048 -keyform PEM
```

To create an RSA private key by using the configuration utility

Navigate to Traffic Management > SSL and, in the SSL Keys group, select Create RSA Key, and create an RSA key.

To create an DSA private key by using the configuration utility

Navigate to Traffic Management > SSL and, in the SSL Keys group, select Create DSA Key, and create a DSA key.

Creating a Certificate Signing Request

The certificate signing request (CSR) is a collection of information, including the domain name, other important company details, and the private key to be used to create the certificate. To avoid generating an invalid certificate, make sure that the details you provide are accurate.

The NetScaler appliance supports creating a CSR signed with the SHA1 digest algorithm by default. In earlier releases, to create a CSR signed with the SHA256 digest algorithm, you had to use OpenSSL.

From release 11.1, the appliance supports creating a CSR signed with the SHA256 digest algorithm. The encryption hash algorithm used in SHA256 makes it stronger than SHA1.

To create a certificate signing request by using the NetScaler command line

At the command prompt, type:

create ssl certreq <reqFile> -**keyFile** <input_filename> **|** -**fipsKeyName** <string>) [-**keyForm** (**DER | PEM**) {-PEMPassPhrase }] -**countryName** <string> -**stateName** <string> -**organizationName** <string> -**organizationUnitName** <string> -**localityName** <string> -**commonName** <string> -**emailAddress** <string> {-challengePassword } -**companyName** <string> -**digestMethod** (**SHA1 | SHA256**)

Example

```
create ssl certreq priv_csr_sha256 -keyfile priv_2048_2 -keyform PEM -
countryName IN -stateName Karnataka -localityName Bangalore -organizationName
Citrix -organizationUnitName NS -digestMethod SHA256
```

To create a certificate signing request by using the NetScaler GUI

1. Navigate to **Traffic Management** > **SSL**.
2. In **SSL Certificate**, click **Create Certificate Signing Request (CSR)**.
3. In **Digest Method**, select **SHA256**.

Submitting the CSR to the CA

Most CAs accept certificate submissions by email. The CA will return a valid certificate to the email address from which you submit the CSR.

Importing Existing Certificates and Keys

If you want to use certificates and keys that you already have on other secure servers or applications in your network, you can export them, and then import them to the NetScaler appliance. You might have to convert exported certificates and keys before you can import them to the NetScaler appliance.

For the details of how to export certificates from secure servers or applications in your network, see the documentation of the server or application from which you want to export.

Note: For installation on the NetScaler appliance, key and certificate names cannot contain spaces or special characters other than those supported by the UNIX file system. Follow the appropriate naming convention when you save the exported key and certificate.

A certificate and private key pair is commonly sent in the PKCS#12 format. The NetScaler supports PEM and DER formats for certificates and keys. To convert PKCS#12 to PEM or DER, or PEM or DER to PKCS#12, see Converting the Format of SSL Certificates for Import or Export.

The NetScaler appliance does not support PEM keys in PKCS#8 format. However, you can convert these keys to a supported format by using the OpenSSL interface, which you can access from the NetScaler command line or the configuration utility. Before you convert the key, you need to verify that the private key is in PKCS#8 format. Keys in PKCS#8 format typically start with the following text:

-----BEGIN ENCRYPTED PRIVATE KEY-----

leuSSZQZKgrgUQ==

-----END ENCRYPTED PRIVATE KEY-----

To open the OpenSSL interface from the command line interface

1. Open an SSH connection to the appliance by using an SSH client, such as PuTTY.
2. Log on to the appliance by using the administrator credentials.
3. At the command prompt, type shell.
4. At the shell prompt type `openssl`.

To open the ssl interface from the configuration utility

Navigate to Traffic Management > SSL and, in the Tools group, select OpenSSL interface.

To convert a non-supported PKCS#8 key format to an encrypted supported key format by using the OpenSSL interface

At the OpenSSL prompt, type one of the following commands, depending on whether the non-supported key format is of type rsa or dsa:

- `rsa`- in <PKCS#8 Key Filename> -des3 -out <encrypted Key Filename>
- `dsa` -in <PKCS#8 Key Filename> -des3 -out <encrypted Key Filename>

To convert a non-supported PKCS#8 key format to an unencrypted key format by using the OpenSSL interface

At the OpenSSL prompt, type the following commands, depending on whether the non-supported key format is of type rsa or dsa:

- `rsa` -in <PKCS#8 Key Filename> -out <unencrypted Key Filename>
- `dsa` -in <PKCS#8 Key Filename> -out <unencrypted Key Filename>

Parameters for converting an unsupported key format to a supported key format

<PKCS#8 Key Filename>
    The input file name of the incompatible PKCS#8 private key.
<encrypted Key Filename>
    The output file name of the compatible encrypted private key in PEM format.
<unencrypted Key Filename>
    The output file name of the compatible unencrypted private key in PEM format.

Generating a Test Certificate

Note

To generate a server test certificate, see Generating a Server Test Certificate.

The NetScaler appliance has a built in CA tools suite that you can use to create self-signed certificates for testing purposes.

Caution: Because these certificates are signed by the NetScaler itself, not by an actual CA, you should not use them in a production environment. If you attempt to use a self-signed certificate in a production environment, users will receive a "certificate invalid" warning each time the virtual server is accessed.

The NetScaler supports creation of the following types of certificates

- Root-CA certificates
- Intermediate-CA certificates
- End-user certificates
  - server certificates
  - client certificates

Before generating a certificate, create a private key and use that to create a certificate signing request (CSR) on the appliance. Then, instead of sending the CSR out to a CA, use the NetScaler CA Tools to generate a certificate.

For details on how to create a private key and a CSR, see Obtaining a Certificate from a Certificate Authority.

To create a certificate by using a wizard

1. Navigate to Traffic Management > SSL.
2. In the details pane, under Getting Started, select the wizard for the type of certificate that you want to create.
3. Follow the instructions on the screen.

To create a Root-CA certificate by using the command line interface

At the command prompt, type the following command:

create ssl cert <certFile> <reqFile> <certType> [-keyFile <input_filename>] [-keyform (**DER | PEM**)] [-days <positive_integer>]

Example

```
In the following example, csreq1 is the CSR and rsa1 is the private key that was created ear
> create ssl cert cert1 csreq1 ROOT_CERT -keyFile rsa1 -keyForm PEM -days 365
  Done
```

To create an Intermediate-CA certificate certificate by using the command line interface

At the command prompt, type the following command:

create ssl cert <certFile> <reqFile> <certType> [-keyFile <input_filename>] [-keyform (**DER** | **PEM**)] [-days <positive_integer>] [-certForm (**DER** | **PEM**)] [-CAcert <input_filename>] [-CAcertForm (**DER** | **PEM**)] [-CAkey <input_filename>] [-CAkeyForm (**DER** | **PEM**)] [-CAserial <output_filename>]

Example

In the following example, csr1 is the CSR created earlier. Cert1 and rsakey1 are the certificate and corresponding key of the self-signed (root-CA) certificate, and pvtkey1 is the private key of the intermediate-CA certificate.

```
> create ssl cert certsy csr1 INTM_CERT -CAcert cert1 -CAkey rsakey1 -CAserial 23
Done
```

```
> create ssl rsakey pvtkey1 2048 -exponent F4 -keyform PEM
```

To create a Root-CA certificate by using the configuration utility

Navigate to Traffic Management > SSL and, in the Getting Started group, select Root-CA Certificate Wizard, and configure a root CA certificate.

To create an Intermediate-CA certificate certificate by using the configuration utility

Navigate to Traffic Management > SSL and, in the Getting Started group, select Intermediate-CA Certificate Wizard, and configure an intermediate CA certificate.

Creating an End-User Certificate

An end-user certificate can be a client certificate or a server certificate. To create a test end-user certificate, specify the Intermediate CA certificate or the self-signed root-CA certificate.

Note: To create an end-user certificate for production use, specify a trusted CA certificate and send the CSR to a certificate authority (CA).

To create a test end-user certificate by using the command line interface

create ssl cert <certFile> <reqFile> <certType> [-keyFile <input_filename>] [-keyform (DER | PEM)] [-days<positive_integer>] [-certForm (DER | PEM)] [-CAcert <input_filename>] [-CAcertForm (DER | PEM)] [-CAkey<input_filename>] [-CAkeyForm (DER | PEM)] [-CAserial <output_filename>]

Example

If there is no intermediate certificate, use the certificate (cert1) and private key (rsakey1) values of the root-CA certificate in CAcert and CAkey.

> create ssl cert cert12 csr1 SRVR_CERT -CAcert cert1 -CAkey rsakey1 -CAserial 23

Done

If there is an intermediate certificate, use the certificate (certsy) and private key (pvtkey1) values of the intermediate certificate in CAcert and CAkey.

> create ssl cert cert12 csr1 SRVR_CERT -CAcert certsy -CAkey pvtkey1 -CAserial 23

Done

Generating a Diffie-Hellman (DH) Key

The Diffie-Hellman (DH) key exchange is a way for two parties involved in an SSL transaction that have no prior knowledge of each other to agree upon a shared secret over an insecure channel. This secret can then be converted into cryptographic keying material for mainly symmetric key cipher algorithms that require such a key exchange.

This feature is disabled by default and should be specifically configured to support ciphers that use DH as the key exchange algorithm.

Note

Generating a 2048-bit DH key may take a long time (up to 30 minutes).

To generate a DH key by using the command line interface

At the command prompt, type the following command:

create ssl dhparam <dhFile> [<bits>] [-gen (2 **|** 5)]

Example

```
create ssl dhparam Key-DH-1 512 -gen 2
```

To generate a DH key by using the configuration utility

Navigate to **Traffic Management** > **SSL** and, in the **Tools** group, select **Create Diffie-Hellman (DH) key**, and generate a DH key.

Adding a Group of SSL Certificates

If the server certificate is issued by an intermediate CA that is not recognized by standard web browsers as a trusted CA, the CA certificate(s) must be sent to the client with the server's own certificate. Otherwise, the browser terminates the SSL session because it fails to authenticate the server certificate.

There are two ways to add the server and intermediate certificates:

- Create a certificate set that contains the chain of certificates.
- Create a chain of certificates manually by adding and linking the certificates individually.

Adding and Linking a Certificate Set

Updated: 2014-06-17

Note: This feature is not supported on the NetScaler FIPS platform.

Instead of adding and linking individual certificates, you can now group a server certificate and up to nine intermediate certificates in a single file, and then specify the file's name when adding a certificate-key pair. Before you do so, make sure that the following prerequisites are met.

- The certificates in the file are in the following order:
  Server certificate (should be the first certificate in the file)
  Optionally, a server key
  Intermediate certificate 1 (ic1)
  Intermediate certificate 2 (ic2)
  Intermediate certificate 3 (ic3), and so on
  Note: Intermediate certificate files are created for each intermediate certificate with the name "<certificatebundlename>.pem_ic<*n*>" where n is between 1 and 9. For example, bundle.pem_ic1, where bundle is the name of the certificate set and ic1 is the first intermediate certificate in the set.
- Bundle option is selected.
- No more than nine intermediate certificates are present in the file.

The file is parsed and the server certificate, intermediate certificates, and server key (if present) are identified. First, the server certificate and key are added. Then, the intermediate certificates are added, in the order in which they were added to the file, and linked accordingly.

An error is reported if any of the following conditions exist:

- A certificate file for one of the intermediate certificates already exists on the appliance.
- The key is placed before the server certificate in the file.
- An intermediate certificate is placed before the server certificate.
- Intermediate certificates are not in placed in the file in the same order as they are created.
- No certificates are present in the file.
- A certificate is not in the proper PEM format.
- The number of intermediate certificates in the file exceeds nine.

To add a certificate set by using the command line interface

At the command prompt, type the following commands to create a certificate set and verify the configuration:

1. add ssl certKey <certkeyName> -cert <string> -key <string> -bundle (YES | NO)
2. show ssl certKey
3. show ssl certlink

Example

In the following example, the certificate set (bundle.pem) contains the following files:

- server certificate (bundle) linked to bundle_ic1
- First intermediate certificate (bundle_ic1) linked to bundle_ic2
- Second intermediate certificate (bundle_ic2) linked to bundle_ic3
- Third intermediate certificate (bundle_ic3)

```
 > add ssl certKey bundle -cert bundle.pem -key bundle.pem -bundle yes

> sh ssl certkey

1)      Name: ns-server-certificate

        Cert Path: ns-server.cert

        Key Path: ns-server.key

        Format: PEM

        Status: Valid,   Days to expiration:5733

        Certificate Expiry Monitor: ENABLED

        Expiry Notification period: 30 days

        Certificate Type: Server Certificate

        Version: 3

        Serial Number: 01

        Signature Algorithm: sha256WithRSAEncryption

        Issuer:  C=US,ST=California,L=San Jose,O=Citrix ANG,OU=NS Internal,CN=default OULLFT

        Validity

                Not Before: Apr 21 15:56:16 2016 GMT

                Not After : Mar  3 06:30:56 2032 GMT

        Subject:  C=US,ST=California,L=San Jose,O=Citrix ANG,OU=NS Internal,CN=default OULLFT

        Public Key Algorithm: rsaEncryption

        Public Key size: 2048

2)      Name: servercert

        Cert Path: complete/server/server_rsa_1024.pem

        Key Path: complete/server/server_rsa_1024.ky

        Format: PEM

        Status: Valid,   Days to expiration:7150

        Certificate Expiry Monitor: ENABLED

        Expiry Notification period: 30 days

        Certificate Type: Server Certificate

        Version: 3

        Serial Number: 1F

        Signature Algorithm: sha1WithRSAEncryption

        Issuer:  C=IN,ST=KAR,O=Citrix R&D Pvt Ltd,CN=Citrix

        Validity

                Not Before: Sep  2 09:54:07 2008 GMT

                Not After : Jan 19 09:54:07 2036 GMT
```

```
      Subject:  C=IN,ST=KAR,O=Citrix Pvt Ltd,CN=Citrix

      Public Key Algorithm: rsaEncryption

      Public Key size: 1024

3)     Name: bundletest

      Cert Path: bundle9.pem

      Key Path: bundle9.pem

      Format: PEM

      Status: Valid,   Days to expiration:3078

      Certificate Expiry Monitor: ENABLED

      Expiry Notification period: 30 days

      Certificate Type: Server Certificate

      Version: 3

      Serial Number: 01

      Signature Algorithm: sha256WithRSAEncryption

      Issuer:  C=IN,ST=ka,O=sslteam,CN=ICA9

      Validity

          Not Before: Nov 28 06:43:11 2014 GMT

          Not After : Nov 25 06:43:11 2024 GMT

      Subject:  C=IN,ST=ka,O=sslteam,CN=Server9

      Public Key Algorithm: rsaEncryption

      Public Key size: 2048

4)     Name: bundletest_ic1

      Cert Path: bundle9.pem_ic1

      Format: PEM

      Status: Valid,   Days to expiration:3078

      Certificate Expiry Monitor: ENABLED

      Expiry Notification period: 30 days

      Certificate Type: Intermediate CA

      Version: 3

      Serial Number: 01

      Signature Algorithm: sha256WithRSAEncryption

      Issuer:  C=IN,ST=ka,O=sslteam,CN=ICA8

      Validity

          Not Before: Nov 28 06:42:56 2014 GMT

          Not After : Nov 25 06:42:56 2024 GMT

      Subject:  C=IN,ST=ka,O=sslteam,CN=ICA9
```

```
        Public Key Algorithm: rsaEncryption

        Public Key size: 2048

5)      Name: bundletest_ic2

        Cert Path: bundle9.pem_ic2

        Format: PEM

        Status: Valid,   Days to expiration:3078

        Certificate Expiry Monitor: ENABLED

        Expiry Notification period: 30 days

        Certificate Type: Intermediate CA

        Version: 3

        Serial Number: 01

        Signature Algorithm: sha256WithRSAEncryption

        Issuer:  C=IN,ST=ka,O=sslteam,CN=ICA7

        Validity

            Not Before: Nov 28 06:42:55 2014 GMT

            Not After : Nov 25 06:42:55 2024 GMT

        Subject:  C=IN,ST=ka,O=sslteam,CN=ICA8

        Public Key Algorithm: rsaEncryption

        Public Key size: 2048

6)      Name: bundletest_ic3

        Cert Path: bundle9.pem_ic3

        Format: PEM

        Status: Valid,   Days to expiration:3078

        Certificate Expiry Monitor: ENABLED

        Expiry Notification period: 30 days

        Certificate Type: Intermediate CA

        Version: 3

        Serial Number: 01

        Signature Algorithm: sha256WithRSAEncryption

        Issuer:  C=IN,ST=ka,O=sslteam,CN=ICA6

        Validity

            Not Before: Nov 28 06:42:53 2014 GMT

            Not After : Nov 25 06:42:53 2024 GMT

        Subject:  C=IN,ST=ka,O=sslteam,CN=ICA7

        Public Key Algorithm: rsaEncryption

        Public Key size: 2048
```

```
7)     Name: bundletest_ic4
       Cert Path: bundle9.pem_ic4
       Format: PEM
       Status: Valid,   Days to expiration:3078
       Certificate Expiry Monitor: ENABLED
       Expiry Notification period: 30 days
       Certificate Type: Intermediate CA
       Version: 3
       Serial Number: 01
       Signature Algorithm: sha256WithRSAEncryption
       Issuer:  C=IN,ST=ka,O=sslteam,CN=ICA5
       Validity
              Not Before: Nov 28 06:42:51 2014 GMT
              Not After : Nov 25 06:42:51 2024 GMT
       Subject:  C=IN,ST=ka,O=sslteam,CN=ICA6
       Public Key Algorithm: rsaEncryption
       Public Key size: 2048
8)     Name: bundletest_ic5
       Cert Path: bundle9.pem_ic5
       Format: PEM
       Status: Valid,   Days to expiration:3078
       Certificate Expiry Monitor: ENABLED
       Expiry Notification period: 30 days
       Certificate Type: Intermediate CA
       Version: 3
       Serial Number: 01
       Signature Algorithm: sha256WithRSAEncryption
       Issuer:  C=IN,ST=ka,O=sslteam,CN=ICA4
       Validity
              Not Before: Nov 28 06:42:50 2014 GMT
              Not After : Nov 25 06:42:50 2024 GMT
       Subject:  C=IN,ST=ka,O=sslteam,CN=ICA5
       Public Key Algorithm: rsaEncryption
       Public Key size: 2048
9)     Name: bundletest_ic6
       Cert Path: bundle9.pem_ic6
```

```
        Format: PEM

        Status: Valid,   Days to expiration:3078

        Certificate Expiry Monitor: ENABLED

        Expiry Notification period: 30 days

        Certificate Type: Intermediate CA

        Version: 3

        Serial Number: 01

        Signature Algorithm: sha256WithRSAEncryption

        Issuer:  C=IN,ST=ka,O=sslteam,CN=ICA3

        Validity

            Not Before: Nov 28 06:42:48 2014 GMT

            Not After : Nov 25 06:42:48 2024 GMT

        Subject:  C=IN,ST=ka,O=sslteam,CN=ICA4

        Public Key Algorithm: rsaEncryption

        Public Key size: 2048

    10)    Name: bundletest_ic7

        Cert Path: bundle9.pem_ic7

        Format: PEM

        Status: Valid,   Days to expiration:3078

        Certificate Expiry Monitor: ENABLED

        Expiry Notification period: 30 days

        Certificate Type: Intermediate CA

        Version: 3

        Serial Number: 01

        Signature Algorithm: sha256WithRSAEncryption

        Issuer:  C=IN,ST=ka,O=sslteam,CN=ICA2

        Validity

            Not Before: Nov 28 06:42:46 2014 GMT

            Not After : Nov 25 06:42:46 2024 GMT

        Subject:  C=IN,ST=ka,O=sslteam,CN=ICA3

        Public Key Algorithm: rsaEncryption

        Public Key size: 2048

    11)    Name: bundletest_ic8

        Cert Path: bundle9.pem_ic8

        Format: PEM

        Status: Valid,   Days to expiration:3078
        Certificate Expiry Monitor: ENABLED

        Expiry Notification period: 30 days

        Certificate Type: Intermediate CA

        Version: 3

        Serial Number: 01

        Signature Algorithm: sha256WithRSAEncryption

        Issuer:  C=IN,ST=ka,O=sslteam,CN=ICA1

        Validity

            Not Before: Nov 28 06:42:45 2014 GMT

            Not After : Nov 25 06:42:45 2024 GMT

        Subject:  C=IN,ST=ka,O=sslteam,CN=ICA2

        Public Key Algorithm: rsaEncryption

        Public Key size: 2048

12)     Name: bundletest_ic9

        Cert Path: bundle9.pem_ic9

        Format: PEM

        Status: Valid,   Days to expiration:3078

        Certificate Expiry Monitor: ENABLED

        Expiry Notification period: 30 days

        Certificate Type: Intermediate CA

        Version: 3

        Serial Number: 01

        Signature Algorithm: sha256WithRSAEncryption

        Issuer:  C=IN,ST=ka,O=sslteam,CN=RootCA4096

        Validity

            Not Before: Nov 28 06:42:43 2014 GMT

            Not After : Nov 25 06:42:43 2024 GMT

        Subject:  C=IN,ST=ka,O=sslteam,CN=ICA1

        Public Key Algorithm: rsaEncryption

        Public Key size: 2048

Done

> sh ssl certlink

1)     Cert Name: bundletest    CA Cert Name: bundletest_ic1

2)     Cert Name: bundletest_ic1        CA Cert Name: bundletest_ic2

3)     Cert Name: bundletest_ic2        CA Cert Name: bundletest_ic3

4)     Cert Name: bundletest_ic3        CA Cert Name: bundletest_ic4
```

```
5)      Cert Name: bundletest_ic4       CA Cert Name: bundletest_ic5

6)      Cert Name: bundletest_ic5       CA Cert Name: bundletest_ic6

7)      Cert Name: bundletest_ic6       CA Cert Name: bundletest_ic7

8)      Cert Name: bundletest_ic7       CA Cert Name: bundletest_ic8

9)      Cert Name: bundletest_ic8       CA Cert Name: bundletest_ic9

Done
```

To add a certificate set by using the configuration utility

1. Navigate to Traffic Management > SSL > Certificates.
2. In the SSL Certificates pane, click Install.
3. In the Install Certificate dialog box, type the details, such as the certificate and key file name, and then select Certificate Bundle.
4. Click Install, and then click Close.

Creating a Chain of Certificates

Updated: 2013-08-20

Instead of using a set of certificates (a single file), you can create a chain of certificates. The chain links the server certificate to its issuer (the intermediate CA). For this approach to work, the intermediate CA certificate file must already be installed on the NetScaler appliance, and one of the certificates in the chain must be trusted by the client application. For example, link Cert-Intermediate-A to Cert-Intermediate-B, where Cert-Intermediate-B is linked to Cert-Intermediate-C, which is a certificate trusted by the client application.

Note: The NetScaler supports sending a maximum of 10 certificates in the chain of certificates sent to the client (one server certificate and nine CA certificates).

To create a certificate chain by using the command line interface

At the command prompt, type the following commands to create a certificate chain and verify the configuration. (Repeat the first command for each new link in the chain.)

- link ssl certkey <certKeyName> <linkCertKeyName>
- show ssl certlink

Example

```
> link ssl certkey siteAcertkey CAcertkey
 Done

> show ssl certlink

linked certificate:
       1) Cert Name: siteAcertkey CA Cert Name: CAcertkey
 Done
```

To create a certificate chain by using the configuration utility

1. Navigate to Traffic Management > SSL > Certificates.
2. Select a server certificate, and in the Action list, select Link, and specify a CA certificate name.

Displaying a Certificate Chain

A certificate contains the name of the issuing authority and the subject to whom the certificate is issued. To validate a certificate, you must look at the issuer of that certificate and confirm if you trust the issuer. If you do not trust the issuer, you must see who issued the issuer certificate. Go up the chain till you reach the root CA certificate or an issuer that you trust.

As part of the SSL handshake, when a client requests a certificate, the NetScaler appliance presents a certificate and the chain of issuer certificates that are present on the appliance. An administrator can view the certificate chain for the certificates present on the appliance and install any missing certificates.

To view the certificate chain for the certificates present on the appliance by using the command line

At the command prompt, type:

show ssl certchain <cert_name>

Examples

There are 3 certificates: c1, c2, and c3. Certificate c1 is signed by c2, c2 is signed by c3, and c3 is the root CA certificate. The following examples illustrate the output of the `show ssl certchain c1` command in different scenarios.

1. Scenario 1:
    - Certificate c2 is linked to c1, and c3 is linked to c2.
    - Certificate c3 is a root CA certificate.

   If you run the following command, the certificate links up to the root CA certificate are displayed.

   ```
   > show ssl certchain c1
   Certificate chain details of certificate name c1 are:
   1) Certificate name: c2               linked; not a root certificate
   2) Certificate name: c3               linked; root certificate
   Done
   ```

2. Scenario 2:
    - Certificate c2 is linked to c1.
    - Certificate c2 is not a root CA certificate.

   If you run the following command, information that certificate c3 is a root CA certificate but is not linked to c2 is displayed.

   ```
   > show ssl certchain c1
   Certificate chain  details of certificate name c1 are:
   1) Certificate Name: c2               linked; not a root certificate
   2) Certificate Name: c3               not linked; root certificate
   Done
   ```

3. Scenario 3:
    - Certificate c1, c2, and c3 are not linked but are present on the appliance.

   If you run the following command, information about all the certificates starting with the issuer of certificate c1 is displayed and it is specified that the certificates are not linked.

   ```
   > show ssl certchain c1
   Certificate chain details of certificate name c1 are:
   1) Certificate Name: c2               not linked; not a root certificate
   2) Certificate Name: c3               not linked; root certificate
   Done
   ```

4. Scenario 4:
    - Certificate c2 is linked to c1.
    - Certificate c3 is not present on the appliance.

   If you run the following command, information about the certificate linked to c1 is displayed and you are prompted to add a certificate with the subject name specified in c2. In this case, the user is asked to add the root CA certificate c3.

   ```
   > show ssl certchain c1
   Certificate chain details of certificate name c1 are:
   1) Certificate Name: c2               linked; not a root certificate
   2) Certificate Name: /C=IN/ST=ka/O=netscaler/CN=test
      Action: Add a certificate with this subject name.
   Done
   ```

5. Scenario 5:
   - A certificate is not linked to certificate c1 and the issuer certificate of c1 is not present on the appliance.

   If you run the following command, you are prompted to add a certificate with the subject name in certificate c1.

   ```
   > sh ssl certchain c1
   Certificate chain details of certificate name c1 are:
   1) Certificate Name: /ST=KA/C=IN
      Action: Add a certificate with this subject name.
   ```

Generating a Server Test Certificate

The NetScaler appliance allows you to create a test certificate for server authentication by using a GUI wizard in the configuration utility. A server certificate is used to authenticate and identify a server in an SSL handshake. A server certificate is generally issued by a trusted CA and is sent out by the server to a client who uses it to authenticate the server.

For issuing a server test certificate, the appliance operates as a CA. This certificate can be bound to an SSL virtual server for authentication in an SSL handshake with a client. This certificate is for testing purposes only. It should not be used in a production environment.

You can install the server test certificate on any virtual server that uses the SSL or the SSL_TCP protocol.

To generate a server test certificate by using the configuration utility

Navigate to Traffic Management > SSL and, in the SSL Certificates group, select Create and Install a Server Test Certificate.

Achieving Perfect Forward Secrecy With DHE

Generating the DH key is a CPU-intensive operation. In earlier releases, key generation, on a VPX appliance, took a long time because it was done in the software. In earlier releases, key generation is optimized (as defined by NIST in http://csrc.nist.gov/publications/nistpubs/800-56A/SP800-56A_Revision1_Mar08-2007.pdf) by setting the dhKeyExpSizeLimit parameter. You can set this parameter for an SSL virtual server or an SSL profile and then bind the profile to a virtual server.

Additionally, to maintain perfect forward secrecy, DH count must ideally be zero. With this enhancement, you can generate a DH key for each transaction (minimum DHcount is 0) without a significant drop in performance, because the operation is optimized. Earlier, the minimum DH count allowed was 500. That is, you could not regenerate the key for up to 500 transactions.

To optimize DH key generation on a VPX appliance by using the command line

At the command prompt, type commands 1 and 2, or type command 3:

1. add ssl profile <name> [-sslProfileType (`BackEnd` | `FrontEnd`)] [-dhCount <positive_integer>] [-dh (`ENABLED` | `DISABLED`) -dhFile <string>] [-dhKeyExpSizeLimit (`ENABLED` | `DISABLED`)]
2. set ssl vserver <vServerName> [-sslProfile <string>]
3. set ssl vserver <vServerName> [-dh (`ENABLED` | `DISABLED`) -dhFile <string>] [-dhCount <positive_integer>] [-dhKeyExpSizeLimit (`ENABLED` | `DISABLED`)]

To optimize DH key generation on a VPX appliance by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers, and open a virtual server.
2. In the SSL Parameters section, select Enable DH Key Expire Size Limit.

Modifying and Monitoring Certificates and Keys

To avoid downtime when replacing a certificate-key pair, you can update an existing certificate. If you want to replace a certificate with a certificate that was issued to a different domain, you must disable domain checks before updating the certificate.

To receive notifications about certificates due to expire, you can enable the expiry monitor.

Updating an Existing Server Certificate

When you remove or unbind a certificate from a configured SSL virtual server, or an SSL service, the virtual server or service becomes inactive until a new valid certificate is bound to it. To avoid downtime, you can use the update feature to replace a certificate-key pair that is bound to an SSL virtual server or an SSL service, without first unbinding the existing certificate.

To update an existing certificate-key pair by using the command line interface

At the command prompt, type the following commands to update an existing certificate-key pair and verify the configuration:

- update ssl certkey <certkeyName> -cert <string> -key <string>
- show ssl certKey <certkeyName>

Example

```
> update ssl certkey siteAcertkey -cert /nsconfig/ssl/cert.pem
       -key /nsconfig/ssl/pkey.pem
  Done

> show ssl certkey siteAcertkey
Name: siteAcertkey       Status: Valid
       Version: 3
       Serial Number: 02
       Signature Algorithm: md5WithRSAEncryption
       Issuer: /C=US/ST=CA/L=Santa Clara/O=siteA/OU=Tech
       Validity
            Not Before: Nov 11 14:58:18 2001 GMT
            Not After: Aug 7 14:58:18 2004 GMT
       Subject: /C=US/ST-CA/L=San Jose/O=CA/OU=Security
       Public Key Algorithm: rsaEncryption
       Public Key size: 2048
  Done
```

To update an existing certificate-key pair by using the configuration utility

Navigate to Traffic Management > SSL > Certificates, select a certificate, and click Update.

Disabling Domain Checks

When an SSL certificate is replaced on the NetScaler, the domain name mentioned on the new certificate should match the domain name of the certificate being replaced. For example, if you have a certificate issued to abc.com, and you are updating it with a certificate issued to def.com, the certificate update fails.

However, if you want the server that has been hosting a particular domain to now host a new domain, you can disable the domain check before updating its certificate.

To disable the domain check for a certificate by using the command line interface

At the command prompt, type the following commands to disable the domain check and verify the configuration:

- update ssl certKey <certkeyName> -noDomainCheck
- show ssl certKey <certkeyName>

Example

```
> update ssl certKey sv -noDomainCheck
```

```
 Done
> show ssl certkey sv
        Name: sv
        Cert Path: /nsconfig/ssl/complete/server/server_rsa_512.pem
        Key Path: /nsconfig/ssl/complete/server/server_rsa_512.ky
        Format: PEM
        Status: Valid,   Days to expiration:9349
        Certificate Expiry Monitor: DISABLED
 Done
```

To disable the domain check for a certificate by using the configuration utility

1. Navigate to Traffic Management > SSL > Certificates, select a certificate, and click Update.
2. Select No Domain Check.

Enabling the Expiry Monitor

An SSL certificate is valid for a specific period of time. A typical deployment includes multiple virtual servers that process SSL transactions, and the certificates bound to them can expire at different times. An expiry monitor configured on the NetScaler appliance creates entries in the appliance's syslog and nsaudit logs when a certificate configured on the appliance is due to expire.

If you want to create SNMP alerts for certificate expiration, you must configure them separately.

To enable an expiry monitor for a certificate by using the command line interface

At the command prompt, type the following commands to enable an expiry monitor for a certificate and verify the configuration:

- set ssl certKey <certkeyName> [-expiryMonitor (ENABLED | DISABLED) [-notificationPeriod <positive_integer>]]
- show ssl certKey <certkeyName>

Example

```
> set ssl certKey sv -expiryMonitor ENABLED â€“notificationPeriod 60
  Done

> show ssl certkey sv
Name: sv
        Cert Path: /nsconfig/ssl/complete/server/server_rsa_512.pem
        Key Path: /nsconfig/ssl/complete/server/server_rsa_512.ky
        Format: PEM
        Status: Valid,   Days to expiration:9349
        Certificate Expiry Monitor: ENABLED
        Expiry Notification period: 60 days
  Done
```

To enable an expiry monitor for a certificate by using the configuration utility

1. Navigate to Traffic Management > SSL > Certificates, select a certificate, and click Update.
2. Select Notify When Expires, and optionally specify a notification period.

Using Global Site Certificates

A global site certificate is a special-purpose server certificate whose key length is greater than 128 bits. A global site certificate consists of a server certificate and an accompanying intermediate-CA certificate. You must import the global site certificate and its key from the server to the NetScaler appliance.

How Global Site Certificates Work

Export versions of browsers use 40-bit encryption to initiate connections to SSL Web-servers. The server responds to connection requests by sending its certificate. The client and server then decide on an encryption strength based on the server certificate type:

- If the server certificate is a normal certificate and not a global site certificate, the export client and server complete the SSL handshake and uses 40-bit encryption for data transfer.
- If the server certificate is a global site certificate (and if the export client feature is supported by the browser) the export client automatically upgrades to 128-bit encryption for data transfer.

If the server certificate is a global site certificate, the server sends its certificate, along with the accompanying intermediate-CA certificate. The browser first validates the intermediate-CA certificate by using one of the Root-CA certificates that are normally included in web browsers. Upon successful validation of the intermediate-CA certificate, the browser uses the intermediate-CA certificate to validate the server certificate. Once the server is successfully validated, the browser renegotiates (upgrades) the SSL connection to 128-bit encryption.

With Microsoft's Server Gated Cryptography (SGC), if the Microsoft IIS server is configured with an SGC certificate, export clients that receive the certificate renegotiate to use128-bit encryption.

Importing a Global Site Certificate

To import a global site certificate, first export the certificate and server key from the Web server. Global site certificates are generally exported in some binary format, therefore, before importing the global site certificate, convert the certificate and key to the PEM format.

To import a global site certificate

1. Using a text editor, copy the server certificate and the accompanying intermediate-CA certificate into two separate files.

   The individual PEM encoded certificate will begin with the header ----- BEGIN CERTIFICATE----- and end with the trailer -----END CERTIFICATE-----.
2. Use an SFTP client to transfer the server certificate, intermediate-CA certificate, and server-key to the NetScaler.
3. Use the following OpenSSL command to identify the server certificate and intermediate-CA certificate from the two separate files.
   Note: You can launch the OpenSSL interface from the configuration utility. In the navigation pane, click SSL. In the details pane, under Tools, click Open SSL interface.

```
> openssl x509 -in >path of the CA cert file<         ï¿½text

X509v3 Basic Constraints:
                CA:TRUE
            X509v3 Key Usage:
                Certificate Sign, CRL Sign
            Netscape Cert Type:
                SSL CA, S/MIME CA


> openssl x509 -in >path of the server certificate file< -text

X509v3 Basic Constraints:
                CA:FALSE
            Netscape Cert Type:
                SSL Server
```

4. At the FreeBSD shell prompt, enter the following command:

```
openssl x509 -in cert.pem -text | more
```

Where **cert.pem** is one of the two certificate files.

Read the **Subject** field in the command output. For example,

```
Subject: C=US, ST=Oregon, L=Portland,
O=mycompany, Inc., OU=IT, CN=www.mycompany.com
```

If the CN field in the Subject matches the domain-name of your Web site, then this is the server certificate and the other certificate is the accompanying intermediate-CA certificate.

5. Use the server certificate and its private key) to create a certificate key pair on the NetScaler. For details on creating a certificate-key pair on the NetScaler, see Adding a Certificate Key Pair.
6. Add the intermediate-CA certificate on the NetScaler. Use the server certificate you created in step 4 to sign this intermediate certificate. For details on creating an Intermediate-CA certificate on the NetScaler, see Generating a Self Signed Certificate.

Converting the Format of SSL Certificates for Import or Export

A NetScaler appliance supports the PEM and DER formats for SSL certificates. Other applications, such as client browsers and some external secure servers, require various public key cryptography standard (PKCS) formats. The NetScaler can convert the PKCS#12 format (the personal information exchange syntax standard) to PEM or DER format for importing a certificate to the appliance, and can convert PEM or DER to PKCS#12 for exporting a certificate. For additional security, conversion of a file for import can include encryption of the private key with the DES or DES3 algorithm.

Note: If you use the configuration utility to import a PKCS#12 certificate, and the password contains a dollar sign ($), backquote (`), or escape (\) character, the import may fail. If it does, the `ERROR: Invalid password` message appears. If you must use a special character in the password, be sure to prefix it with an escape character (\) unless all imports are performed by using the NetScaler command line.

To convert the format of a certificate by using the command line interface

At the command prompt, type the following command:

Convert ssl pkcs12 <outfile> [-import [-pkcs12File <inputFilename>] [-des | -des3] [-export [-certFile <inputFilename>] [-keyFile <inputFilename>]] During the operation, you are prompted to enter an import password or an export password. For a encrypted file, you are also prompted to enter a passphrase.

Example

```
convert ssl pkcs12 Cert-Import-1.pem -import -pkcs12File Cert-Import-1.pfx -des

convert ssl pkcs12 Cert-Client-1.pfx -export -certFile Cert-Client-1 -keyFile Key-Client-1
```

To convert the format of a certificate by using the configuration utility

Navigate to Traffic Management > SSL and, in the Tools group, select Import PKCS#12 or Export PKCS#12.

Enabling Stricter Control on Client Certificate Validation

The NetScaler appliance accepts valid Intermediate-CA certificates if they are issued by a single Root-CA. That is, if only the Root-CA certificate is bound to the virtual server, and any intermediate certificate sent with the client certificate is validated by that Root-CA, the appliance trusts the certificate chain and the handshake is successful.

However, if a client sends a chain of certificates in the handshake, none of the intermediate certificates can be validated by using a CRL or OCSP responder unless that certificate is bound to the SSL virtual server. Therefore, even if one of the intermediate certificates is revoked, the handshake is successful. As part of the handshake, the SSL virtual server sends the list of CA certificates that are bound to it. For stricter control, you can configure the SSL virtual server to accept only a certificate that is signed by one of the CA certificates bound to that virtual server. To do so, you must enable the ClientAuthUseBoundCAChain setting in the SSL profile bound to the virtual server. The handshake fails if the client certificate is not signed by one of the CA certificates bound to the virtual server.

For example, say two client certificates, clientcert1 and clientcert2, are signed by the intermediate certificates Int-CA-A and Int-CA-B, respectively. The intermediate certificates are signed by the root certificate Root-CA. Int-CA-A and Root-CA are bound to the SSL virtual server. In the default case (ClientAuthUseBoundCAChain disabled), both clientcert1 and clientcert2 are accepted. However, ifClientAuthUseBoundCAChain is enabled, only clientcert1 is accepted by the NetScaler appliance

To enable stricter control on client certificate validation by using the command line

At the NetScaler command prompt, type:

```
set ssl profile <name> -ClientAuthUseBoundCAChain Enabled
```

To enable stricter control on client certificate validation by using the configuration utility

1. Navigate to **System** > **Profiles**, select the **SSL Profiles** tab, and create an SSL profile, or select an existing profile.
2. Select **Enable Client Authentication using bound CA Chain**.

Managing Certificate Revocation Lists

A certificate issued by a CA typically remains valid until its expiration date. However, in some circumstances, the CA may revoke the issued certificate before the expiration date (for example, when an owner's private key is compromised, a company's or individual's name changes, or the association between the subject and the CA changes).

A Certificate Revocation List (CRL) identifies invalid certificates by serial number and issuer.

Certificate authorities issue CRLs on a regular basis. You can configure the NetScaler appliance to use a CRL to block client requests that present invalid certificates.

If you already have a CRL file from a CA, add that to the NetScaler. You can configure refresh options. You can also configure the NetScaler to sync the CRL file automatically at a specified interval, from either a web location or an LDAP location. The NetScaler supports CRLs in either the PEM or the DER file format. Be sure to specify the file format of the CRL file being added to the NetScaler.

If you have used the NetScaler as a CA to create certificates that are used in SSL deployments, you can also create a CRL to revoke a particular certificate. This feature can be used, for example, to ensure that self-signed certificates that are created on the NetScaler are not used either in a production environment or beyond a particular date.

Note:

By default, CRLs are stored in the /var/netscaler/ssl directory on the NetScaler appliance.

To manage certificate revocation lists, see the following sections:

- Creating a CRL on the NetScaler
- Adding an Existing CRL to the NetScaler
- Configuring CRL Refresh Parameters
- Synchronizing CRLs
- Performing Client Authentication by using a Certificate Revocation List

Creating a CRL on the NetScaler

Updated: 2013-09-04

Since you can use the NetScaler appliance to act as a certificate authority and create self-signed certificates, you can also revoke certificates that you have created and certificates whose CA certificate you own.

The appliance must revoke invalid certificates before creating a CRL for those certificates. The appliance stores the serial numbers of revoked certificates in an index file and updates the file each time it revokes a certificate. The index file is automatically created the first time a certificate is revoked.

To revoke a certificate or create a CRL by using the command line interface

At the command prompt, type the following command:

create ssl crl <CAcertFile> <CAkeyFile> <indexFile> (-revoke <input_filename> | -genCRL <output_filename>)

Example

```
create ssl crl Cert-CA-1 Key-CA-1 File-Index-1 -revoke Invalid-1

create ssl crl Cert-CA-1 Key-CA-1 File-Index-1 -genCRL CRL-1
```

To revoke a certificate or create a CRL by using the configuration utility

1. Navigate to Traffic Management > SSL and, in the Getting Started group, select CRL Management.
2. Enter the certificate details and, in the Choose Operation list, select Revoke Certificate or Generate CRL.

Adding an Existing CRL to the NetScaler

Updated: 2013-09-05

Before you configure the CRL on the NetScaler appliance, make sure that the CRL file is stored locally on the NetScaler. In the case of an HA setup, the CRL file must be present on both NetScaler appliances, and the directory path to the file must be the same on both appliances.

To add a CRL on the NetScaler by using the command line

At the command prompt, type the following commands to add a CRL on the NetScaler and verify the configuration:

- add ssl crl <crlName> <crlPath> [-inform (DER | PEM)]
- show ssl crl [<crlName>]

Example

```
> add ssl crl crl-one /var/netscaler/ssl/CRL-one -inform PEM
 Done
> show ssl crl crl-one
        Name: crl-one   Status: Valid,  Days to expiration: 29
        CRL Path: /var/netscaler/ssl/CRL-one
        Format: PEM     CAcert: samplecertkey
        Refresh: DISABLED
        Version: 1
        Signature Algorithm: sha1WithRSAEncryption
        Issuer:  C=US,ST=California,L=Santa Clara,O=NetScaler Inc.,OU=SSL Acceleration,CN=ww
        Last_update:Jun 15 10:53:53 2010 GMT
        Next_update:Jul 15 10:53:53 2010 GMT

1)      Serial Number: 00
        Revocation Date:Jun 15 10:51:16 2010 GMT
 Done
```

To add a CRL on the NetScaler by using the configuration utility

Navigate to Traffic Management > SSL > CRL, and add a CRL.

Configuring CRL Refresh Parameters

Updated: 2014-09-29

A CRL is generated and published by a Certificate Authority periodically or, in some cases, immediately after a particular certificate is revoked. Citrix recommends that you update CRLs on the NetScaler appliance regularly, for protection against clients trying to connect with certificates that are not valid.

The NetScaler can refresh CRLs from a web location or an LDAP directory. When you specify refresh parameters and a web location or an LDAP server, the CRL does not have to be present on the local hard disk drive at the time you execute the command. The first refresh stores a copy on the local hard disk drive, in the path specified by the CRL File parameter. The default path for storing the CRL is /var/netscaler/ssl.
Note: In release 10.0 and later, the method for refreshing a CRL is not included by default. You must explicitly specify an HTTP or LDAP method. If you are upgrading from an earlier release to release 10.0 or later, you must add a method and run the command again.

To configure CRL autorefresh by using the command line

At the command prompt, type the following commands to configure CRL auto refresh and verify the configuration the following commands to configure CRL auto refresh and verify the configuration:

- set ssl crl <crlName> [-refresh (ENABLED | DISABLED)] [-CAcert <string>] [-url <URL | -server <ip_addr|ipv6_addr>] [-method HTTP | (LDAP [-baseDN <string>] [-bindDN <string>] [-scope (Base | One)] [-password <string>] [-binary (YES | NO)])] [-port <port>] [-interval <interval>]
- show ssl crl [<crlName>]

Example

```
Set CRL crl1  -refresh enabled -method ldap -inform DER -CAcert ca1 -server 10.102.192.192 -

set ssl crl crl1 -refresh enabled -method http -cacert ca1 -port 80 -time 00:10 -url http://

> sh crl

1)         Name: crl1        Status: Valid,     Days to expiration: 355
            CRL Path: /var/netscaler/ssl/crl1
            Format: PEM      CAcert: ca1
            Refresh: ENABLED          Method: HTTP
            URL: http://10.102.192.192/crl/ca1.crl                 Port:80
            Refresh Time: 00:10
            Last Update: Successful, Date:Tue Jul  6 14:38:13 2010
Done
```

To configure CRL autorefresh using LDAP or HTTP by using the configuration utility

1. Navigate to Traffic Management > SSL > CRL.
2. Open a CRL, and select Enable CRL Auto Refresh.
   Note: If the new CRL has been refreshed in the external repository before its actual update time as specified by the Last Update time field of the CRL, you should immediately refresh the CRL on the NetScaler.

   To view the last update time, select the CRL, and click Details.

Synchronizing CRLs

Updated: 2013-09-03

The NetScaler appliance uses the most recently distributed CRL to prevent clients with revoked certificates from accessing secure resources.

If CRLs are updated often, the NetScaler needs an automated mechanism to fetch the latest CRLs from the repository. You can configure the NetScaler to update CRLs automatically at a specified refresh interval.

The NetScaler maintains an internal list of CRLs that need to be updated at regular intervals. At these specified intervals, the appliance scans the list for CRLs that need to be updated, connects to the remote LDAP server or HTTP server, retrieves the latest CRLs, and then updates the local CRL list with the new CRLs.

Note: If CRL check is set to mandatory when the CA certificate is bound to the virtual server, and the initial CRL refresh fails, all client-authentication connections with the same issuer as the CRL are rejected as REVOKED until the CRL is successfully refreshed.

You can specify the interval at which the CRL refresh should be carried out. You can also specify the exact time.

To synchronize CRL autorefresh by using the command line interface

At the command prompt, type the following command:

set ssl crl <crlName> [-interval <interval>] [-day <integer>] [-time <HH:MM>]

Example

```
set ssl crl CRL-1 -refresh ENABLE -interval
MONTHLY -days 10 -time 12:00
```

To synchronize CRL refresh by using the configuration utility

1. Navigate to Traffic Management > SSL > CRL.
2. Open a CRL, select enable CRL Auto Refresh, and specify the interval.

Performing Client Authentication by using a Certificate Revocation List

Updated: 2013-09-03

If a certificate revocation list (CRL) is present on a NetScaler appliance, a CRL check is performed regardless of whether performing the CRL check is set to mandatory or optional.
The success or failure of a handshake depends on a combination of the following factors:

- Rule for CRL check
- Rule for client certificate check
- State of the CRL configured for the CA certificate

The following table lists the results of the possible combinations for a handshake involving a revoked certificate.

Table 1. Result of a Handshake with a Client Using a Revoked Certificate

Rule for CRL Check	Rule for Client Certificate Check	State of the CRL Configured for the CA certificate	Result of a Handshake with a Revoked Certificate
Optional	Optional	Missing	Success
Optional	Mandatory	Missing	Success
Optional	Mandatory	Present	Failure

Mandatory	Optional	Missing	Success
Mandatory	Mandatory	Missing	Failure
Mandatory	Optional	Present	Success
Mandatory	Mandatory	Present	Failure
Optional/Mandatory	Optional	Expired	Success
Optional/Mandatory	Mandatory	Expired	Failure

Note:

- The CRL check is optional by default. To change from optional to mandatory or vice-versa, you must first unbind the certificate from the SSL virtual server, and then bind it again after changing the option.

- In the output of the `sh ssl vserver` command, `OCSP check: optional` implies that a CRL check is also optional. The CRL check settings are displayed in the output of the `sh ssl vserver` command only if CRL check is set to mandatory. If CRL check is set to optional, the CRL check details do not appear.

To configure CRL check by using the command line interface

At the command prompt, type the following command:

```
bind ssl vserver <vServerName> -certkeyName <string> [(-CA -crlCheck ( Mandatory | Optional
```

Example

```
bind ssl vs v1 -certkeyName ca -CA -crlCheck mandatory
sh ssl vserver
> sh ssl vs v1

Advanced SSL configuration for VServer v1:

DH: DISABLED
Ephemeral RSA: ENABLED Refresh Count: 0
Session Reuse: ENABLED Timeout: 120 seconds
Cipher Redirect: DISABLED
SSLv2 Redirect: DISABLED
ClearText Port: 0
Client Auth: ENABLED Client Cert Required: Mandatory
SSL Redirect: DISABLED
Non FIPS Ciphers: DISABLED
SNI: DISABLED
SSLv2: DISABLED SSLv3: ENABLED TLSv1: ENABLED
Push Encryption Trigger: Always
Send Close-Notify: YES

1) CertKey Name: ca CA Certificate CRLCheck: Mandatory CA_Name Sent

1) Cipher Name: DEFAULT
Description: Predefined Cipher Alias
Done
```

To configure CRL check by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers, and open an SSL virtual server.
2. Click in the Certificates section.
3. Select a certificate and, in the OCSP and CRL Check list, select CRL Mandatory.

Monitoring Certificate Status with OCSP

Online Certificate Status Protocol (OCSP) is an Internet protocol that is used to determine the status of a client SSL certificate. NetScaler appliances support OCSP as defined in RFC 2560. OCSP offers significant advantages over certificate revocation lists (CRLs) in terms of timely information. Up-to-date revocation status of a client certificate is especially useful in transactions involving large sums of money and high-value stock trades. It also uses fewer system and network resources. NetScaler implementation of OCSP includes request batching and response caching.

To monitor certificate status with OCSP, see the following sections:

- NetScaler Implementation of OCSP
- OCSP Request Batching
- OCSP Response Caching
- Configuring an OCSP Responder

NetScaler Implementation of OCSP

OCSP validation on a NetScaler appliance begins when the appliance receives a client certificate during an SSL handshake. To validate the certificate, the NetScaler creates an OCSP request and forwards it to the OCSP responder. To do so, the NetScaler uses a locally configured URL. The transaction is in a suspended state until the NetScaler evaluates the response from the server and determines whether to allow the transaction or reject it. If the response from the server is delayed beyond the configured time and no other responders are configured, the NetScaler will allow the transaction or display an error, depending on whether the OCSP check was set to optional or mandatory, respectively.

The NetScaler supports batching of OCSP requests and caching of OCSP responses to reduce the load on the OCSP responder and provide faster responses.

OCSP Request Batching

Each time the NetScaler receives a client certificate, it sends a request to the OCSP responder. To help avoid overloading the OCSP responder, the NetScaler can query the status of more than one client certificate in the same request. For this to work efficiently, a timeout needs to be defined so that processing of a single certificate is not inordinately delayed while waiting to form a batch.

OCSP Response Caching

Caching of responses received from the OCSP responder enables faster responses to the clients and reduces the load on the OCSP responder. Upon receiving the revocation status of a client certificate from the OCSP responder, the NetScaler caches the response locally for a predefined length of time. When a client certificate is received during an SSL handshake, the NetScaler first checks its local cache for an entry for this certificate. If an entry is found that is still valid (within the cache timeout limit), it is evaluated and the client certificate is accepted or rejected. If a certificate is not found, the NetScaler sends a request to the OCSP responder and stores the response in its local cache for a configured length of time.

Configuring an OCSP Responder

Configuring OCSP involves adding an OCSP responder, binding the OCSP responder to a certification authority (CA) certificate, and binding the certificate to an SSL virtual server. If you need to bind a different certificate to an OCSP responder that has already been configured, you need to first unbind the responder and then bind the responder to a different certificate.

To add an OCSP responder by using the command line interface

At the command prompt, type the following commands to configure OCSP and verify the configuration:

- add ssl ocspResponder <name> -url <URL> [-cache (ENABLED | DISABLED)[-cacheTimeout <positive_integer>]] [-batchingDepth <positive_integer>][-batchingDelay <positive_integer>] [-resptimeout <positive_integer>] [-responderCert <string> | -trustResponder] [-producedAtTimeSkew <positive_integer>][-signingCert <string>][-useNonce (YES | NO)][-insertClientCert(YES | NO)]
- bind ssl certKey [<certkeyName>] [-ocspResponder <string>] [-priority <positive_integer>]
- bind ssl vserver <vServerName>@ (-certkeyName <string> (CA [-ocspCheck (Mandatory | Optional)]))
- show ssl ocspResponder [<name>]

Example

```
add ssl ocspResponder ocsp_responder1 -url "http:// www.myCA.org:80/ocsp/" -cache ENABLED -c
bind ssl certKey ca_cert -ocspResponder ocsp_responder1 -priority 1
bind ssl vserver vs1 -certkeyName ca_cert -CA -ocspCheck Mandatory

sh ocspResponder ocsp_responder1
1)Name: ocsp_responder1
URL: http://www.myCA.org:80/ocsp/, IP: 192.128.22.22
Caching: Enabled        Timeout: 30 minutes
Batching: 8 Timeout: 100 mS
HTTP Request Timeout: 100mS
Request Signing Certificate: sign_cert
Response Verification: Full, Certificate: responder_cert
ProducedAt Time Skew: 300 s
Nonce Extension: Enabled
 Client Cert Insertion: Enabled
Done

show certkey ca_cert
Name: ca_cert     Status: Valid,   Days to expiration:8907
Version: 3
â€¦
1)  VServer name: vs1      CA Certificate
1)  OCSP Responder name: ocsp_responder1     Priority: 1
Done

sh ssl vs vs1
Advanced SSL configuration for VServer vs1:
DH: DISABLED
â€¦
1) CertKey Name: ca_cert CA Certificate OCSPCheck: Mandatory
1) Cipher Name: DEFAULT
  Description: Predefined Cipher Alias
Done
```

To modify an OCSP responder by using the command line interface

You cannot modify the responder name. All other parameters can be changed using the set ssl ocspResponder command.

At the command prompt, type the following commands to set the parameters and verify the configuration:

- set ssl ocspResponder <name> [-url <URL>] [-cache (ENABLED | DISABLED)] [-cacheTimeout <positive_integer>] [-batchingDepth <positive_integer>] [-batchingDelay <positive_integer>] [-resptimeout <positive_integer>] [-responderCert <string> | -trustResponder][-producedAtTimeSkew <positive_integer>][- signingCert <string>] [-useNonce (YES | NO)]
- unbind ssl certKey [<certkeyName>] [-ocspResponder <string>]
- bind ssl certKey [<certkeyName>] [-ocspResponder <string>] [-priority <positive_integer>]
- show ssl ocspResponder [<name>]

To configure an OCSP responder by using the configuration utility

1. Navigate to Traffic Management > SSL > OCSP Responder, and configure an OCSP responder.
2. Navigate to Traffic Management > SSL > Certificates, select a certificate, and in the Action list, select OCSP Bindings. Bind an OCSP responder.
3. Navigate to Traffic Management > Load Balancing > Virtual Servers, open a virtual server, and click in the Certificates section to bind a CA certificate.
4. Optionally, select select OCSP Mandatory.

Configuring Client Authentication

In a typical SSL transaction, the client that is connecting to a server over a secure connection checks the validity of the server by checking the server's certificate before initiating the SSL transaction. In some cases, however, you might want to configure the server to authenticate the client that is connecting to it.

With client authentication enabled on an SSL virtual server, the NetScaler appliance asks for the client certificate during the SSL handshake. The appliance checks the certificate presented by the client for normal constraints, such as the issuer signature and expiration date.

Note: For the NetScaler to verify issuer signatures, the certificate of the CA that issued the client certificate must be installed on the NetScaler and bound to the virtual server that the client is transacting with.

If the certificate is valid, the NetScaler allows the client to access all secure resources. But if the certificate is invalid, the NetScaler drops the client request during the SSL handshake.

The NetScaler verifies the client certificate by first forming a chain of certificates, starting with the client certificate and ending with the root CA certificate for the client (for example, VeriSign). The root CA certificate may contain one or more intermediate CA certificates (if the client certificate is not directly issued by the root CA).

Before you enable client authentication on the NetScaler, make sure that a valid client certificate is installed on the client. Then, enable client authentication for the virtual server that will handle the transactions. Finally, bind the certificate of the CA that issued the client certificate to the virtual server on the NetScaler.

Note: A NetScaler MPX appliance supports a certificate-key pair size from 512 to 4096 bits. The certificate must be signed by using one of the following hash algorithms:

- MD5
- SHA-1
- SHA-224
- SHA-256
- SHA-384
- SHA-512

On an SDX appliance, if an SSL chip is assigned to a VPX instance, the certificate-key pair size support of an MPX appliance applies. Otherwise, the normal certificate-key pair size support of a VPX instance applies.

A NetScaler virtual appliance (VPX instance) supports certificates of at least 512 bits, up to the following sizes:

- 4096-bit server certificate on the virtual server
- 4096-bit client certificate on the service
- 4096-bit CA certificate
- 2048-bit certificate on the physical server
- 2048-bit client certificate (if client authentication is enabled on the virtual server)

To configure client authentication, see the following sections:

- Providing the Client Certificate
- Enabling Client-Certificate-Based Authentication
- Binding CA Certificates to the Virtual Server

Providing the Client Certificate

Before you configure client authentication, a valid client certificate must installed on the client. A client certificate includes details about the specific client system that will create secure sessions with the NetScaler appliance. Each client certificate is unique and should be used by only one client system.

Whether you obtain the client certificate from a CA, use an existing client certificate, or generate a client certificate on the NetScaler appliance, you must convert the certificate to the correct format. On the NetScaler, certificates are stored in either the PEM or DER format and must be converted to PKCS#12 format before they are installed on the client system. After converting the certificate and transferring it to the client, system, make sure that it is installed on that system and configured for the client application that will be part of the SSL transactions (for example, the web browser).

For instructions on how to convert a certificate from PEM or DER format to PKCS#12 format, see Converting SSL Certificates for Import or Export.

For instructions on how to generate a client certificate, see Generating Self-Signed Certificates.

Enabling Client-Certificate-Based Authentication

Updated: 2013-08-20

By default, client authentication is disabled on the NetScaler appliance, and all SSL transactions proceed without authenticating the client. You can configure client authentication to be either optional or mandatory as part of the SSL handshake.

If client authentication is optional, the NetScaler requests the client certificate but proceeds with the SSL transaction even if the client presents an invalid certificate. If client authentication is mandatory, the NetScaler terminates the SSL handshake if the SSL client does not provide a valid certificate.

Caution: Citrix recommends that you define proper access control policies before changing client-certificate-based authentication check to optional.
Note: Client authentication is configured for individual SSL virtual servers, not globally.

To enable client-certificate-based authentication by using the command line interface

At the command prompt, type the following commands to enable the client-certificate-based authentication and verify the configuration:

- set ssl vserver <vServerName> [-clientAuth (ENABLED | DISABLED)] [-clientCert (MANDATORY | OPTIONAL)]
- show ssl vserver <vServerName>

Example

```
> set ssl vserver vssl -clientAuth ENABLED -clientCert Mandatory
 Done
> show ssl vserver vssl

        Advanced SSL configuration for VServer vssl:
        DH: DISABLED
        Ephemeral RSA: ENABLED          Refresh Count: 0
        Session Reuse: ENABLED          Timeout: 120 seconds
        Cipher Redirect: DISABLED
        SSLv2 Redirect: DISABLED
        ClearText Port: 0
        Client Auth: ENABLED    Client Cert Required: Mandatory
        SSL Redirect: DISABLED
        Non FIPS Ciphers: DISABLED
        SSLv2: DISABLED SSLv3: ENABLED  TLSv1: ENABLED

1)      CertKey Name: sslckey   Server Certificate

1)      Policy Name: client_cert_policy  Priority: 0

1)      Cipher Name: DEFAULT
        Description: Predefined Cipher Alias
 Done
```

To enable client-certificate-based authentication by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers, and open a virtual server.
2. In the SSL Parameters section, select Client Authentication, and in the Client Certificate list, select Mandatory.

Binding CA Certificates to the Virtual Server

A CA whose certificate is present on the NetScaler appliance must issue the client certificate used for client authentication. You must bind this certificate to the NetScaler virtual server that will carry out client authentication.

You must bind the CA certificate to the SSL virtual server in such a way that the NetScaler can form a complete certificate chain when it verifies the client certificate. Otherwise, certificate chain formation fails and the client is denied access even if its certificate is valid.

You can bind CA certificates to the SSL virtual server in any order. The NetScaler forms the proper order during client certificate verification.

For example, if the client presents a certificate issued by **CA_A**, where **CA_A** is an intermediate CA whose certificate is issued by **CA_B**, whose certificate is in turn issued by a trusted root CA, **Root_CA**, a chain of certificates that contain all three of these certificates must be bound to the virtual server on the NetScaler.

For instructions on binding one or more certificates to the virtual server, see Binding the Certificate-key Pair to the SSL Based Virtual Server.

For instructions on creating a chain of certificates, see Creating a Chain of Certificates.

Customizing the SSL Configuration

Once your basic SSL configuration is operational, you can customize some of the parameters that are specific to the certificates being used in SSL transactions. You can also enable and disable session reuse and client authentication, and you can configure redirect responses for cipher and SSLv2 protocol mismatches.

You can also customize SSL settings for two NetScaler appliances in a High Availability configuration, and you can synchronize settings, certificates and keys across those appliances.

These settings will depend on your network deployment and the type of clients you expect will connect to your servers.

To customize the SSL configuration, see the following sections:

- Configuring Diffie-Hellman (DH) Parameters
- Configuring Ephemeral RSA
- Configuring Session Reuse
- Configuring Cipher Redirection
- Configuring SSLv2 Redirection
- Configuring SSL Protocol Settings
- Configuring Close-Notify
- Configuring ECDHE Ciphers
- Leveraging Hardware and Software to Improve ECDHE Cipher Performance
- ECDSA Cipher Suites support on MPX and SDX appliances with N3 chips
- Configuring a Common Name on an SSL Service or Service Group for Server Certificate Authentication
- Configuring Advanced SSL Settings
- Synchronizing Configuration Files in a High Availability Setup
- Managing Server Authentication
- Configuring User-Defined Cipher Groups on the NetScaler Appliance

Configuring Diffie-Hellman (DH) Parameters

If you are using ciphers on the NetScaler that require a DH key exchange to set up the SSL transaction, enable DH key exchange on the NetScaler and configure other settings based on your network.

To list the ciphers for which DH parameters must be set by using the NetScaler command line, type: `sh cipher DH`.

To list the ciphers for which DH parameters must be set by using the configuration utility, navigate to Traffic Management > SSL > Cipher Groups, and double-click DH.

For details on how to enable DH key exchange, see Generating a Diffie-Hellman (DH) Key.

To configure DH Parameters by using the command line interface

At the command prompt, type the following commands to configure DH parameters and verify the configuration:

- set ssl vserver <vserverName> -dh <Option> -dhCount <RefreshCountValue> -filepath <string>
- show ssl vserver <vServerName>

Example

```
> set ssl vserver vs-server -dh ENABLED  -dhFile /nsconfig/ssl/ns-server.cert -dhCount 1000
  Done
> show ssl vserver vs-server

        Advanced SSL configuration for VServer vs-server:
        DH: ENABLED
        Ephemeral RSA: ENABLED          Refresh Count: 1000
        Session Reuse: ENABLED          Timeout: 120 seconds
        Cipher Redirect: DISABLED
        SSLv2 Redirect: DISABLED
        ClearText Port: 0
        Client Auth: DISABLED
        SSL Redirect: DISABLED
        Non FIPS Ciphers: DISABLED
        SSLv2: DISABLED SSLv3: ENABLED  TLSv1: ENABLED

1)      Cipher Name: DEFAULT
        Description: Predefined Cipher Alias
 Done
```

To configure DH Parameters by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers, and open a virtual server.
2. In the SSL Parameters section, select Enable DH Param, and specify a refresh count and file path.

Configuring Ephemeral RSA

Ephemeral RSA allows export clients to communicate with the secure server even if the server certificate does not support export clients (1024-bit certificate). If you want to prevent export clients from accessing the secure web object and/or resource, you need to disable ephemeral RSA key exchange.

By default, this feature is enabled on the NetScaler appliance, with the refresh count set to zero (infinite use).

Note:

The ephemeral RSA key is automatically generated when you bind an export cipher to an SSL or TCP-based SSL virtual server or service. When you remove the export cipher, the eRSA key is not deleted but reused at a later date when another export cipher is bound to an SSL or TCP-based SSL virtual server or service. The eRSA key is deleted when the system restarts.

To configure Ephemeral RSA by using the command line interface

At the command prompt, type the following commands to configure ephemeral RSA and verify the configuration:

- set ssl vserver <vServerName> -eRSA (enabled | disabled) -eRSACount <positive_integer>
- show ssl vserver <vServerName>

Example

```
> set ssl vserver vs-server -eRSA ENABLED -eRSACount 1000
 Done
> show ssl vserver vs-server

        Advanced SSL configuration for VServer vs-server:
        DH: DISABLED
        Ephemeral RSA: ENABLED          Refresh Count: 1000
        Session Reuse: ENABLED          Timeout: 120 seconds
        Cipher Redirect: DISABLED
        SSLv2 Redirect: DISABLED
        ClearText Port: 0
        Client Auth: DISABLED
        SSL Redirect: DISABLED
        Non FIPS Ciphers: DISABLED
        SSLv2: DISABLED SSLv3: ENABLED  TLSv1: ENABLED

1)      Cipher Name: DEFAULT
        Description: Predefined Cipher Alias
 Done
```

To configure Ephemeral RSA by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers, and open a virtual server.
2. In the SSL Parameters section, select Enable Ephemereal RSA, and specify a refresh count.

Configuring Session Reuse

For SSL transactions, establishing the initial SSL handshake requires CPU-intensive public key encryption operations. Most handshake operations are associated with the exchange of the SSL session key (client key exchange message). When a client session is idle for some time and is then resumed, the SSL handshake is typically conducted all over again. With session reuse enabled, session key exchange is avoided for session resumption requests received from the client.

Session reuse is enabled on the NetScaler appliance by default. Enabling this feature reduces server load, improves response time, and increases the number of SSL transactions per second (TPS) that can be supported by the server.

To configure session reuse by using the command line interface

At the command prompt, type the following commands to configure session reuse and verify the configuration:

- set ssl vserver <vServerName> -sessReuse (ENABLED | DISABLED) -sessTimeout <positive_integer>
- show ssl vserver <vServerName>

Example

```
> set ssl vserver vs-ssl -sessreuse enabled -sesstimeout 600
 Done

> show ssl vserver vs-ssl

        Advanced SSL configuration for VServer vs-ssl:
        DH: DISABLED
        Ephemeral RSA: ENABLED          Refresh Count: 1000
        Session Reuse: ENABLED          Timeout: 600 seconds
        Cipher Redirect: DISABLED
        SSLv2 Redirect: DISABLED
        ClearText Port: 0
        Client Auth: DISABLED
        SSL Redirect: DISABLED
        Non FIPS Ciphers: DISABLED
        SSLv2: DISABLED SSLv3: ENABLED  TLSv1: ENABLED

1)      CertKey Name: Auth-Cert-1       Server Certificate

1)      Cipher Name: DEFAULT
        Description: Predefined Cipher Alias
 Done
```

To configure session reuse by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers, and open a virtual server.
2. In the SSL Parameters section, select Enable Session Reuse, and specify a time for which to keep the session active.

Configuring Cipher Redirection

During the SSL handshake, the SSL client (usually a web browser) announces the suite of ciphers that it supports, in the configured order of cipher preference. From that list, the SSL server then selects a cipher that matches its own list of configured ciphers.

If the ciphers announced by the client do not match those configured on the SSL server, the SSL handshake fails, and the failure is announced by a cryptic error message displayed in the browser. These messages rarely mention the exact cause of the error.

With cipher redirection, you can configure an SSL virtual server to deliver accurate, meaningful error messages when an SSL handshake fails. When SSL handshake fails, the NetScaler appliance redirects the user to a previously configured URL or, if no URL is configured, displays an internally generated error page.

To configure cipher redirection by using the command line interface

At the command prompt, type the following commands to configure cipher redirection and verify the configuration:

- set ssl vserver <vServerName> -cipherRedirect < ENABLED | DISABLED> -cipherURL < URL>
- show ssl vserver <vServerName>

Example

```
> set ssl vserver vs-ssl -cipherRedirect ENABLED -cipherURL http://redirectURl
 Done
> show ssl vserver vs-ssl

        Advanced SSL configuration for VServer vs-ssl:
        DH: DISABLED
        Ephemeral RSA: ENABLED          Refresh Count: 1000
        Session Reuse: ENABLED          Timeout: 600 seconds
        Cipher Redirect: ENABLED        Redirect URL: http://redirectURl
        SSLv2 Redirect: DISABLED
        ClearText Port: 0
        Client Auth: DISABLED
        SSL Redirect: DISABLED
        Non FIPS Ciphers: DISABLED
        SSLv2: DISABLED SSLv3: ENABLED  TLSv1: ENABLED

1)      CertKey Name: Auth-Cert-1       Server Certificate

1)      Cipher Name: DEFAULT
        Description: Predefined Cipher Alias
 Done
```

To configure cipher redirection by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers, and open a virtual server.
2. In the SSL Parameters section, select Enable Cipher Redirect, and specify a redirect URL.

Configuring SSLv2 Redirection

For an SSL transaction to be initiated, and for successful completion of the SSL handshake, the server and the client should agree on an SSL protocol that both of them support. If the SSL protocol version supported by the client is not acceptable to the server, the server does not go ahead with the transaction, and an error message is displayed.

You can configure the server to display a precise error message (user-configured or internally generated) advising the client on the next action to be taken. Configuring the server to display this message requires that you set up SSLv2 redirection.

To configure SSLv2 redirection by using the command line interface

At the command prompt, type the following commands to configure SSLv2 redirection and verify the configuration:

- set ssl vserver <vServerName> [-sslv2Redirect (ENABLED | DISABLED) [-sslv2URL <URL>]]
- show ssl vserver <vServerName>

Example

```
> set ssl vserver vs-ssl -sslv2Redirect ENABLED -sslv2URL http://sslv2URL
 Done
> show ssl vserver vs-ssl

        Advanced SSL configuration for VServer vs-ssl:
        DH: DISABLED
        Ephemeral RSA: ENABLED          Refresh Count: 1000
        Session Reuse: ENABLED          Timeout: 600 seconds
        Cipher Redirect: DISABLED
        SSLv2 Redirect: ENABLED Redirect URL: http://sslv2URL
        ClearText Port: 0
        Client Auth: DISABLED
        SSL Redirect: DISABLED
        Non FIPS Ciphers: DISABLED
        SSLv2: DISABLED SSLv3: ENABLED  TLSv1: ENABLED

1)      CertKey Name: Auth-Cert-1       Server Certificate

1)      Cipher Name: DEFAULT
        Description: Predefined Cipher Alias
 Done
```

To configure SSLv2 redirection by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers, and open a virtual server.
2. In the SSL Parameters section, select SSLv2 Redirect, and specify a URL.

Configuring SSL Protocol Settings

The NetScaler appliance supports the SSLv2, SSLv3, TLSv1, TLSv1.1, and TLSv1.2 protocols. Each of these can be set on the appliance as required by your deployment and the type of clients that will connect to the appliance.

TLS protocol versions 1.0, 1.1, and 1.2 are more secure than older versions of the TLS/SSL protocol. However, to support legacy systems, many TLS implementations maintain backward compatibility with the SSLv3 protocol. In an SSL handshake, the highest protocol version common to the client and the SSL virtual server configured on the NetScaler appliance is used.

In the first handshake attempt, a TLS client offers the highest protocol version that it supports. If the handshake fails, the client offers a lower protocol version. For example, if a handshake with TLS version 1.1 is not successful, the client attempts to renegotiate by offering the TLSv1.0 protocol. If that attempt is unsuccessful, the client reattempts with the SSLv3 protocol. A â€œman in the middleâ€• (MITM) attacker can break the initial handshake and trigger renegotiation with the SSLv3 protocol, and then exploit a vulnerability in SSLv3. To mitigate such attacks, you can disable SSLv3 or not allow renegotiation using a downgraded protocol. However, this might not be practical if your deployment includes legacy systems. An alternative is to recognize a signaling cipher suite value (TLS_FALLBACK_SCSV) in the client request.

A TLS_FALLBACK_SCSV value in a client hello message indicates to the virtual server that the client has previously attempted to connect with a higher protocol version and that the current request is a fallback. If the virtual server detects this value, and it supports a version higher than the one indicated by the client, it rejects the connection with a fatal alert. If a TLS_FALLBACK_SCSV is not included in the client hello message, or if the protocol version in the client hello is the highest protocol version supported by the virtual server, the handshake succeeds.

To configure SSL protocol support by using the command line interface

At the command prompt, type the following commands to configure SSL protocol support and verify the configuration:

- set ssl vserver <vServerName> -ssl2 (ENABLED | DISABLED) -ssl3 (ENABLED | DISABLED) -tls1 (ENABLED | DISABLED) -tls11 (ENABLED | DISABLED) -tls12 (ENABLED | DISABLED)
- show ssl vserver <vServerName>

Example

```
> set ssl vserver vs-ssl -tls11 ENABLED -tls12 ENABLED
 Done
> sh ssl vs vs-ssl

                                          Advanced SSL configuration for VServer vs-ssl:
        DH: DISABLED
        Ephemeral RSA: ENABLED                          Refresh Count: 0
        Session Reuse: ENABLED                            Timeout: 120 seconds
        Cipher Redirect: DISABLED
        SSLv2 Redirect: DISABLED
        ClearText Port: 0
        Client Auth: DISABLED
        SSL Redirect: DISABLED
        Non FIPS Ciphers: DISABLED
        SNI: DISABLED
        SSLv2: DISABLED        SSLv3: ENABLED    TLSv1.0: ENABLED  TLSv1.1: ENABLED  TLSv1.2
        Push Encryption Trigger: Always
        Send Close-Notify: YES

        1 bound certificate:
1)      CertKey Name: mycert  Server Certificate

        1 configured cipher:
1)      Cipher Name: DEFAULT
        Description: Predefined Cipher Alias
Done
```

To configure SSL protocol support by using the configuration Utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers, and open a virtual server.
2. In the SSL Parameters section, select a protocol to enable.

Configuring Close-Notify

A close-notify is a secure message that indicates the end of SSL data transmission. A close-notify setting is required at the global level. This setting applies to all virtual servers, services, and service groups. For information about the global setting, see Configuring Advanced SSL Settings.

In addition to the global setting, you can set the close-notify parameter at the virtual server, service, or service group level. You therefore have the flexibility of setting the parameter for one entity and unsetting it for another entity. However, make sure that you set this parameter at the global level. Otherwise, the setting at the entity level does not apply.

To configure close-notify at the entity level by using the command line interface

At the command prompt, type any of the following commands to configure close-notify and verify the configuration:

1. To configure close-notify at the virtual server level, type:
   - set ssl vserver <vServerName> -sendCloseNotify (**YES | NO**)
   - show ssl vserver <vServerName>
2. To configure close-notify at the service level, type:
   - set ssl service <serviceName> -sendCloseNotify (**YES | NO**)
   - show ssl service <serviceName>
3. To configure close-notify at the service group level, type:
   - set ssl serviceGroup <serviceGroupName> -sendCloseNotify (**YES | NO**)
   - show ssl serviceGroup <serviceGroupName>

Example

```
> set ssl vserver sslvsvr -sendCloseNotify YES
 Done
```

To configure close-notify at the entity level by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers, and open a virtual server.
2. In the SSL Parameters section, select Send Close-Notify.

Configuring ECDHE Ciphers

The Citrix NetScaler VPX, MPX, SDX, and FIPS appliances support the ECDHE cipher group in release 10.5 build 53.9 or later. On an SDX appliance, if an SSL chip is assigned to a VPX instance, the cipher support of an MPX appliance applies. Otherwise, the normal cipher support of a VPX instance applies. For a complete list of ciphers supported by the NetScaler appliance, see Ciphers Supported by the NetScaler Appliance.

For a list of ECDHE ciphers supported on NetScaler appliances, see Cipher/Protocol Support Matrix on the NetScaler Appliance.

ECDHE cipher suites use elliptical curve cryptography (ECC). Because of its smaller key size, ECC is especially useful in a mobile (wireless) environment or an interactive voice response environment, where every millisecond is important. Smaller key sizes save power, memory, bandwidth, and computational cost.

A NetScaler appliance supports the following ECC curves:

- P_256
- P_384
- P_224
- P_521

Note: If you upgrade from a build earlier than release 10.1 build 121.10, you must explicitly bind ECC curves to your existing SSL virtual servers or front end services. The curves are bound by default to any virtual servers or front end services that you create after the upgrade.

You can bind an ECC curve to SSL front-end entities only. By default all four curves are bound, in the following order: P_256, P_384,P_224,P_521. To change the order, you must first unbind all the curves, and then bind them in the desired order.

To unbind ECC curves and bind an ECC curve to an SSL virtual server by using the command line

At the command prompt, type:

- unbind ssl vserver <vServerName> -eccCurveName ALL
- bind ssl vserver <vServerName> -eccCurveName <eccCurveName>

Example

```
unbind ssl vs v1 â€“eccCurvename ALL
bind ssl vserver  v1 -eccCurveName P_224
> sh ssl vserver v1

        Advanced SSL configuration for VServer v1:
        DH: DISABLED
        Ephemeral RSA: ENABLED              Refresh Count: 0
        Session Reuse: ENABLED              Timeout: 120 seconds
        Cipher Redirect: DISABLED
        SSLv2 Redirect: DISABLED
        ClearText Port: 0
        Client Auth: DISABLED
        SSL Redirect: DISABLED
        Non FIPS Ciphers: DISABLED
        SNI: DISABLED
        SSLv2: DISABLED       SSLv3: ENABLED       TLSv1.0: ENABLED  TLSv1.1: DISABLED  TL
        Push Encryption Trigger: Always
        Send Close-Notify: YES

        ECC Curve: P_224

1)        Cipher Name: DEFAULT
        Description: Predefined Cipher Alias
 Done
```

Leveraging Hardware and Software to Improve ECDHE Cipher Performance

Note

This enhancement is applicable only to the MPX/SDX 11542, MPX/SDX 14000, MPX 22000, MPX 24000, and MPX 25000 series appliances.

Previously, ECDHE computation on a NetScaler appliance was performed only on the hardware (Cavium chips), which limited the number of SSL sessions at any given time. With this enhancement, some operations are also performed in the software. That is, processing is done both on the Cavium chips and on the CPU cores to improve ECDHE cipher performance.

The processing is first performed in software, up to the configured software crypto threshold. After this threshold is reached, the operations are offloaded to the hardware. Therefore, this hybrid model leverages both hardware and software to improve SSL performance. You can enable the hybrid model by setting the â€œ softwareCryptoThresholdâ€• parameter to suit your requirement. To disable the hybrid model, set this parameter to 0.

Benefits are greatest if the current CPU utilization is not too high, because the CPU threshold is not exclusive to ECDHE computation. For example, if the current workload on the NetScaler appliance consumes 50% of the CPU cycles, and the threshold is set to 80%, ECDHE computation can use an additional 30% of the cycles. After the configured software crypto threshold of 80% is reached, further ECDHE computation is offloaded to the hardware. In that case, actual CPU utilization might exceed 80%, because performing ECDHE computations in hardware consumes some CPU cycles.

To enable the hybrid model by using the NetScaler command line

At the NetScaler command prompt, type:

set ssl parameter -softwareCryptoThreshold <positive_integer>

Synopsis

softwareCryptoThreshold:

NetScaler CPU utilization threshold (as a percentage) beyond which crypto operations are not done in software. A value of zero implies that CPU is not utilized for doing crypto in software.

Default = 0

Min = 0

Max = 100

Example

```
>set ssl parameter - softwareCryptoThreshold 80

Done

>show ssl parameter

Advanced SSL Parameters


SSL quantum size                                      : 8 KB

Max CRL memory size                                   : 256 MB

Strict CA checks                                      : NO

Encryption trigger timeout                            : 100 ms

Send Close-Notify                                     : YES
```

```
  Encryption trigger packet count                       : 45

  Deny SSL Renegotiation                                : ALL

  Subject/Issuer Name Insertion Format                  : Unicode

  OCSP cache size                                       : 10 MB

  Push flag                                             : 0x0 (Auto)

  Strict Host Header check for SNI enabled SSL sessions : NO

  PUSH encryption trigger timeout                       : 1 ms

  Crypto Device Disable Limit                           : 0

  Global undef action for control policies              : CLIENTAUTH

  Global undef action for data policies                 : NOOP

  Default profile                                       : DISABLED

  Disable TLS 1.1/1.2 for SSL_BRIDGE secure monitors    : NO

  Disable TLS 1.1/1.2 for dynamic and VPN services      : NO

  Software Crypto acceleration CPU Threshold            : 80

  Signature and Hash Algorithms supported by TLS1.2     : ALL
```

To enable the hybrid model by using the NetScaler GUI

1. Navigate to **Traffic Management** > **SSL** > **Change advanced SSL settings**.
2. Enter a value for **Software Crypto Threshold (%)**.

ECDSA Cipher Suites support on MPX and SDX appliances with N3 chips

The NetScaler MPX and SDX appliances now support the elliptical curve digital signature algorithm (ECDSA) cipher group. In ECDHE_ECDSA, the server's certificate must contain an ECDSA-capable public key. For client authentication, an ECDSA CA certificate must be bound to the virtual server.

To support ECDSA cipher suites on a NetScaler SDX appliance, an SSL core must be assigned to the VPX instance.

ECDSA cipher suites use elliptical curve cryptography (ECC). Because of its smaller size, it is particularly helpful in environments where processing power, storage space, bandwidth, and power consumption are constrained.

Note

if you add an ECDSA certificate-key pair, only the following curves are supported:

- prime256v1
- secp384r1

The following table lists the ECDSA ciphers that are supported on the NetScaler MPX and SDX appliances with N3 chips:

Cipher Name	Priority	Description	Key Exchange Algorithm	Authentication Algorithm	En Alg Si
TLS1-ECDHE-ECDSA-AES128-SHA	1	SSLv3	ECC-DHE	ECDSA	AE
TLS1-ECDHE-ECDSA-AES256-SHA	2	SSLv3	ECC-DHE	ECDSA	AE
TLS1.2-ECDHE-ECDSA-AES128-SHA256	3	TLSv1.2	ECC-DHE	ECDSA	AE
TLS1.2-ECDHE-ECDSA-AES256-SHA384	4	TLSv1.2	ECC-DHE	ECDSA	AE
TLS1.2-ECDHE-ECDSA-AES128-GCM-SHA256	5	TLSv1.2	ECC-DHE	ECDSA	AE (1
TLS1.2-ECDHE-ECDSA-AES256-GCM-SHA384	6	TLSv1.2	ECC-DHE	ECDSA	AE (2
TLS1-ECDHE-ECDSA-RC4-SHA	7	SSLv3	ECC-DHE	ECDSA	R
TLS1-ECDHE-ECDSA-DES-CBC3-SHA	8	SSLv3	ECC-DHE	ECDSA	3D

Important

ECDSA Cipher group support is available only at the front end, on MPX and SDX appliances with N3 chips. Use the **show ns hardware** command to find out if your appliance has N3 chips.

Example

```
> sh ns hardware

          Platform: NSMPX-22000 16*CPU+24*IX+12*E1K+2*E1K+4*CVM N3 2200100

          Manufactured on: 8/19/2013

          CPU: 2900MHZ

          Host Id: 1006665862

          Serial no: ENUK6298FT

          Encoded serial no: ENUK6298FT

 Done
```

Configuring a Common Name on an SSL Service or Service Group for Server Certificate Authentication

In end-to-end encryption with server authentication enabled, you can include a common name in the configuration of an SSL service or service group. The name that you specify is compared to the common name in the server certificate during an SSL handshake. If the two names match, the handshake is successful. If the common names do not match, the common name specified for the service or service group is compared to values in the subject alternative name (SAN) field in the certificate. If it matches one of those values, the handshake is successful. This configuration is especially useful if there are, for example, two servers behind a firewall and one of the servers spoofs the identity of the other. If the common name is not checked, a certificate presented by either server is accepted if the IP address matches.
Note: Only domain name, URL, and email ID DNS entries in the SAN field are compared.

To configure common-name verification for an SSL service or service group by using the command line interface

At the command prompt, type the following commands to specify server authentication with common-name verification and verify the configuration:

1. To configure common name in a service, type:
   - set ssl service <serviceName> -commonName <string> -serverAuth ENABLED
   - show ssl service <serviceName>
2. To configure common name in a service group, type:
   - set ssl serviceGroup <serviceGroupName> -commonName <string> -serverAuth ENABLED
   - show ssl serviceGroup <serviceGroupName>

Example

```
> set ssl service svc1 -commonName xyz.com -serverAuth ENABLED
 Done
> show ssl service svc1
        Advanced SSL configuration for Back-end SSL Service svc1:
        DH: DISABLED
        Ephemeral RSA: DISABLED
        Session Reuse: ENABLED                Timeout: 300 seconds
        Cipher Redirect: DISABLED
        SSLv2 Redirect: DISABLED
        Server Auth: ENABLED        Common Name: www.xyz.com
        SSL Redirect: DISABLED
        Non FIPS Ciphers: DISABLED
        SNI: DISABLED
        SSLv2: DISABLED        SSLv3: ENABLED        TLSv1: ENABLED

1)        CertKey Name: cacert        CA Certificate                OCSPCheck: Optional

1)        Cipher Name: ALL
        Description: Predefined Cipher Alias
 Done
```

To configure common-name verification for an SSL service or service group by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Services or Navigate to Traffic Management > Load Balancing > Service Groups, and open a service or service group.
2. In the SSL Parameters section, select Enable Server Authentication, and specify a common name.

Configuring Advanced SSL Settings

Advanced customization of your SSL configuration addresses specific issues. You can use the set ssl parameter command o the configuration utility to specify the following:

- Quantum size to be used for SSL transactions.
- CRL memory size.
- OCSP cache size.
- Deny SSL renegotiation.
- Set the PUSH flag for decrypted, encrypted, or all records.
- Drop requests if the client initiates the handshake for one domain and sends an HTTP request for another domain.
- Set the time after which encryption is triggered.
  Note: The time that you specify applies only if you use the set ssl vserver command or the configuration utility to set timer-based encryption.

To configure advanced SSL settings by using the command line interface

At the command prompt, type the following commands to configure advanced SSL settings and verify the configuration:

- set ssl parameter [-quantumSize <quantumSize>] [-crlMemorySizeMB <positive_integer>] [-strictCAChecks (YES | NO)] [-sslTriggerTimeout <positive_integer>] [-sendCloseNotify (YES | NO)] [-encryptTriggerPktCoun <positive_integer>] [-denySSLReneg <denySSLReneg>] [-insertionEncoding (Unicode|UTF-8)] [-ocspCacheSize <positive_integer>][- pushFlag <positive_integer>] [- dropReqWithNoHostHeader (YES | NO)] [-pushEncTriggerTimeout <positive_integer>]
- show ssl parameter

Example

```
> set ssl parameter -quantumSize 8 -crlMemorySizeMB 256 -strictCAChecks no -sslt
iggerTimeout 100 -sendClosenotify no -encryptTriggerPktCount 45 -denySSLReneg no
-insertionEncoding unicode -ocspCacheSize 10 -pushFlag 3 -dropReqWithNoHostHeader YES  -push
 Done

> show ssl parameter
Advanced SSL Parameters
-----------------------
        SSL quantum size:               8 kB
        Max CRL memory size:            256 MB
        Strict CA checks:               NO
        Encryption trigger timeout      100 mS
        Send Close-Notify               NO
        Encryption trigger packet count:        45
        Deny SSL Renegotiation          NO
        Subject/Issuer Name Insertion Format:   Unicode
        OCSP cache size:        10 MB
               Push flag:      0x3 (On every decrypted and encrypted record)
                                                                Strict Host Header check for
                                                                 PUSH encryption trigger tim
 Done
```

To configure advanced SSL settings by using the configuration utility

Navigate to Traffic Management > SSL and, in the Settings group, select Change advanced SSL settings.

PUSH Flag-Based Encryption Trigger Mechanism

The encryption trigger mechanism that is based on the PSH TCP flag now enables you to do the following:

- Merge consecutive packets in which the PSH flag is set into a single SSL record, or ignore the PSH flag.
- Perform timer-based encryption, in which the time-out value is set globally by using the set ssl parameter -pushEncTriggerTimeout <positive_integer> command.

To configure PUSH flag-based encryption by using the command line interface

At the command prompt, type the following commands to configure PUSH flag-based encryption and verify the configuration:

- set ssl vserver <vServerName> [-pushEncTrigger <pushEncTrigger>]
- show ssl vserver

Example

```
Advanced SSL configuration for VServer v1:
                DH: DISABLED
                Ephemeral RSA: ENABLED                              Refresh Count: 0
                Session Reuse: ENABLED                                Timeout: 120 seconds
                Cipher Redirect: DISABLED
                SSLv2 Redirect: DISABLED
                ClearText Port: 0
                Client Auth: DISABLED
                SSL Redirect: DISABLED
                Non FIPS Ciphers: DISABLED
                SNI: DISABLED
                SSLv2: DISABLED               SSLv3: ENABLED                TLSv1: ENABLED
                Push Encryption Trigger: Always
```

To configure PUSH flag-based encryption by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual servers and open an SSL virtual server.
2. In the SSL Parameters section, from the PUSH Encryption Trigger list, select a value.

Synchronizing Configuration Files in a High Availability Setup

In a high availability (HA) set up, the primary NetScaler appliance in the HA pair automatically synchronizes with the secondary appliance in the pair. In the synchronization process, the secondary copies the primary's /nsconfig/ssl/ directory, which is the default location for storing the certificates and keys for SSL transactions. Synchronization occurs at one-minute intervals and every time a new file is added to the directory.

To synchronize files in a high availability setup by using the command line interface

At the command prompt, type the following command:

sync HA files [<Mode>]

Example

```
sync HA files SSL
```

To synchronize files in a high availability setup by using the configuration utility

Navigate to Traffic Management > SSL and, in the Tools group, select Start SSL certificate, key file synchronization for HA.

Managing Server Authentication

Since the NetScaler appliance performs SSL offload and acceleration on behalf of a web server, the appliance does not usually authenticate the Web server's certificate. However, you can authenticate the server in deployments that require end-to-end SSL encryption.

In such a situation, the NetScaler becomes the SSL client, carries out a secure transaction with the SSL server, verifies that a CA whose certificate is bound to the SSL service has signed the server certificate, and checks the validity of the server certificate.

To authenticate the server, you must first enable server authentication and then bind the certificate of the CA that signed the server's certificate to the SSL service on the NetScaler. When binding the certificate, you must specify the bind as CA option.

To enable (or disable) server certificate authentication by using the command line interface

At the command prompt, type the following commands to enable server certificate authentication and verify the configuration:

- set ssl service <serviceName> -serverAuth (ENABLED | DISABLED)
- show ssl service <serviceName>

Example

```
> set ssl service ssl-service-1 -serverAuth ENABLED
 Done
> show ssl service ssl-service-1

        Advanced SSL configuration for Back-end SSL Service ssl-service-1:
        DH: DISABLED
        Ephemeral RSA: DISABLED
        Session Reuse: ENABLED          Timeout: 300 seconds
        Cipher Redirect: DISABLED
        SSLv2 Redirect: DISABLED
        Server Auth: ENABLED
        SSL Redirect: DISABLED
        Non FIPS Ciphers: DISABLED
        SSLv2: DISABLED SSLv3: ENABLED  TLSv1: ENABLED

1)      Cipher Name: ALL
        Description: Predefined Cipher Alias
 Done
```

To enable (or disable) server certificate authentication by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Services, and open an SSL service.
2. In the SSL Parameters section, select Enable Server Authentication, and specify a Common Name.
3. In Advanced Settings, select Certificates, and bind a CA certificate to the service.

To bind the CA certificate to the service by using the command line interface

At the command prompt, type the following commands to bind the CA certificate to the service and verify the configuration:

- bind ssl service <serviceName> -certkeyName <string> -CA
- show ssl service <serviceName>

Example

```
> bind ssl service ssl-service-1 -certkeyName samplecertkey -CA
 Done
> show ssl service ssl-service-1
```

```
        Advanced SSL configuration for Back-end SSL Service ssl-service-1:
        DH: DISABLED
        Ephemeral RSA: DISABLED
        Session Reuse: ENABLED          Timeout: 300 seconds
        Cipher Redirect: DISABLED
        SSLv2 Redirect: DISABLED
        Server Auth: ENABLED
        SSL Redirect: DISABLED
        Non FIPS Ciphers: DISABLED
        SSLv2: DISABLED SSLv3: ENABLED  TLSv1: ENABLED

1)      CertKey Name: samplecertkey     CA Certificate          CRLCheck: Optional

1)      Cipher Name: ALL
        Description: Predefined Cipher Alias
 Done
```

Configuring User-Defined Cipher Groups on the NetScaler Appliance

A cipher group is a set of cipher suites that you bind to an SSL virtual server, service, or service group on the NetScaler appliance. A cipher suite comprises a protocol, a key exchange (Kx) algorithm, an authentication (Au) algorithm, an encryption (Enc) algorithm, and a message authentication code (Mac) algorithm. Your appliance ships with a predefined set of cipher groups. When you create a SSL service or SSL service group, the ALL cipher group is automatically bound to it. However, when you create an SSL virtual server or a transparent SSL service, the DEFAULT cipher group is automatically bound to it. In addition, you can create a user-defined cipher group and bind it to an SSL virtual server, service, or service group.
Note: If your MPX appliance does not have any licenses, then only the EXPORT cipher is bound to your SSL virtual server, service, or service group.

To create a user-defined cipher group, first you create a cipher group and then you bind ciphers or cipher groups to this group. If you specify a cipher alias or a cipher group, all the ciphers in the cipher alias or group are added to the user-defined cipher group. You can also add individual ciphers (cipher suites) to a user-defined group. However, you cannot modify a predefined cipher group. Before removing a cipher group, unbind all the cipher suites in the group.

If you bind a cipher group to an SSL virtual server, service, or service group, the ciphers are appended to the existing ciphers that are bound to the entity. To bind a specific cipher group to the entity, you must first unbind the ciphers or cipher group tha is bound to the entity and then bind the specific cipher group. For example, to bind only the AES cipher group to an SSL service, you perform the following steps:

1. Unbind the default cipher group ALL that is bound by default to the service when the service is created.

   ```
   unbind ssl service <service name> -cipherName ALL
   ```
2. Bind the AES cipher group to the service

   ```
   bind ssl service <Service name> -cipherName AE
   ```

If you want to bind the cipher group DES in addition to AES, at the command prompt, type:

- `bind ssl service <service name> -cipherName DES`

Note: The free NetScaler virtual appliance supports only the DH cipher group.

To configure a user-defined cipher group by using the command line interfac

At the command prompt, type the following commands to add a cipher group, or to add ciphers to a previously created group, and verify the settings:

- add ssl cipher <cipherGroupName>
- bind ssl cipher <cipherGroupName> -cipherName <string>
- show ssl cipher <cipherGroupName>

Example

```
 > add ssl cipher test
 Done
> bind ssl cipher test -cipherName SSLv2
 Done
> show ssl cipher test
1)      Cipher Name: SSL2-RC2-CBC-MD5
Description: SSLv2 Kx=RSA      Au=RSA  Enc=RC2(128)  Mac=MD5
2)      Cipher Name: SSL2-RC4-MD5
Description: SSLv2 Kx=RSA      Au=RSA  Enc=RC4(128)  Mac=MD5
3)      Cipher Name: SSL2-DES-CBC3-MD5
Description: SSLv2 Kx=RSA      Au=RSA  Enc=3DES(168) Mac=MD5
4)      Cipher Name: SSL2-DES-CBC-MD5
Description: SSLv2 Kx=RSA      Au=RSA  Enc=DES(56)   Mac=MD5
5)      Cipher Name: SSL2-RC4-64-MD5
Description: SSLv2 Kx=RSA      Au=RSA  Enc=RC4(64)   Mac=MD5
6)      Cipher Name: SSL2-EXP-RC4-MD5
Description: SSLv2 Kx=RSA(512) Au=RSA  Enc=RC4(40)   Mac=MD5  Export
7)      Cipher Name: SSL2-EXP-RC2-CBC-MD5
Description: SSLv2 Kx=RSA(512) Au=RSA  Enc=RC2(40)   Mac=MD5  Export
 Done
```

To unbind ciphers from a cipher group by using the command line interface

At the command prompt, type the following commands to unbind ciphers from a user-defined cipher group, and verify the settings:

- show ssl cipher <cipherGroupName>
- unbind ssl cipher <cipherGroupName> -cipherName <string>
- show ssl cipher <cipherGroupName>

Example

```
> show ssl cipher test
1) Cipher Name: SSL2-RC2-CBC-MD5
Description: SSLv2 Kx=RSA Au=RSA Enc=RC2(128) Mac=MD5
2) Cipher Name: SSL2-RC4-MD5
Description: SSLv2 Kx=RSA Au=RSA Enc=RC4(128) Mac=MD5
3) Cipher Name: SSL2-DES-CBC3-MD5
Description: SSLv2 Kx=RSA Au=RSA Enc=3DES(168) Mac=MD5
4) Cipher Name: SSL2-DES-CBC-MD5
Description: SSLv2 Kx=RSA Au=RSA Enc=DES(56) Mac=MD5
5) Cipher Name: SSL2-RC4-64-MD5
Description: SSLv2 Kx=RSA Au=RSA Enc=RC4(64) Mac=MD5
6) Cipher Name: SSL2-EXP-RC4-MD5
Description: SSLv2 Kx=RSA(512) Au=RSA Enc=RC4(40) Mac=MD5 Export
7) Cipher Name: SSL2-EXP-RC2-CBC-MD5
Description: SSLv2 Kx=RSA(512) Au=RSA Enc=RC2(40) Mac=MD5 Export
Done

> unbind ssl cipher test -cipherName SSL2-RC2-CBC-MD5

> show ssl cipher test
1) Cipher Name: SSL2-RC4-MD5
Description: SSLv2 Kx=RSA Au=RSA Enc=RC4(128) Mac=MD5
2) Cipher Name: SSL2-DES-CBC3-MD5
Description: SSLv2 Kx=RSA Au=RSA Enc=3DES(168) Mac=MD5
3) Cipher Name: SSL2-DES-CBC-MD5
Description: SSLv2 Kx=RSA Au=RSA Enc=DES(56) Mac=MD5
4) Cipher Name: SSL2-RC4-64-MD5
Description: SSLv2 Kx=RSA Au=RSA Enc=RC4(64) Mac=MD5
5) Cipher Name: SSL2-EXP-RC4-MD5
Description: SSLv2 Kx=RSA(512) Au=RSA Enc=RC4(40) Mac=MD5 Export
6) Cipher Name: SSL2-EXP-RC2-CBC-MD5
Description: SSLv2 Kx=RSA(512) Au=RSA Enc=RC2(40) Mac=MD5 Export
Done
```

To remove a cipher group by using the command line interface

Note: You cannot remove a built-in cipher group. Before removing a user-defined cipher group, make sure that the cipher group is empty.

At the command prompt, type the following commands to remove a user-defined cipher group, and verify the configuration:

- rm ssl cipher <userDefCipherGroupName> [<cipherName> ...]
- show ssl cipher <cipherGroupName>

Example

```
> rm ssl cipher test
Done

> sh ssl cipher test
ERROR: No such resource [cipherGroupName, test]
```

To configure a user-defined cipher group by using the configuration utility

Navigate to Traffic Management > SSL > Cipher Groups, and configure a cipher group.

To bind a cipher group to an SSL virtual server, service, or service group by using the command line interface

At the command prompt, type one of the following:

- bind ssl vserver <vServerName> -cipherName <string>
- bind ssl service <serviceName> -cipherName <string>
- bind ssl serviceGroup <serviceGroupName> -cipherName <string>

Examples

```
> bind ssl vserver ssl_vserver_test -cipherName test
Done
bind ssl service  nshttps -cipherName test
Done
> bind ssl servicegroup  ssl_svc  -cipherName test
Done
```

To bind a cipher group to an SSL virtual server, service, or service group by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers or navigate to Traffic Management > Load Balancing > Services or navigate to Traffic Management > Load Balancing > Service Groups, and open the virtual server, service, or service group.
2. In Advanced Settings, select SSL Ciphers, and bind a cipher group to the virtual server, service, or service group.

Configuring SSL Actions and Policies

An SSL policy evaluates incoming traffic and applies a predefined action to requests that match a rule (expression). You have to configure the actions before creating the policies, so that you can specify an action when you create a policy. To put a policy into effect, you must either bind it to a virtual server on the appliance, so that it applies only to traffic flowing through that virtual server, or bind it globally, so that it applies to all traffic flowing through the appliance.

SSL actions define SSL settings that you can apply to the selected requests. You associate an action with one or more policies. Data in client connection requests or responses is compared to a rule specified in the policy, and the action is applied to connections that match the rule (expression).

You can configure classic policies with classic expressions and default syntax policies with default syntax expressions for SSL.
Note: Users who are not experienced in configuring policies at the NetScaler command line usually find using the configuration utility to be considerably easier.

You can associate a user-defined action or a built-in action to a default syntax policy. Classic policies allow only user-defined actions. In default syntax policy, you can also group policies under a policy label, in which case they are applied only when invoked from another policy.

Common uses of SSL actions and policies include per-directory client authentication, support for Outlook web access, and SSL-based header insertions. SSL-based header insertions contain SSL settings required by a server whose SSL processing has been offloaded to the NetScaler appliance.

To configure SSL actions and policies, see the following sections:

- Configuring User-Defined Actions for SSL Policies
- Configuring SSL Policies
- Configuring an SSL Default Syntax Policy
- Configuring Built-in Actions for SSL Default Syntax Policies
- Configuring SSL Policy Labels
- Configuring Per-Directory Client Authentication
- Configuring Support for Outlook Web Access
- Configuring SSL-Based Header Insertion
- Binding SSL Policies to a Virtual Server
- Binding SSL Policies Globally

Configuring User-Defined Actions for SSL Policies

SSL policies require that you create an action before creating a policy, so that you can specify the actions when you create the policies. In SSL default syntax policies, you can also use the built-in actions. For more information about built-in actions, see Configuring Built-in SSL Actions.

To configure an SSL action by using the command line interface

At the command prompt, type the following commands to configure an action and verify the configuration:

- add SSL action <name> -clientAuth(DOCLIENTAUTH | NOCLIENTAUTH) -clientCert (ENABLED | DISABLED) certHeader <string> -clientHeader <string> -clientCertSerialNumber (ENABLED | DISABLED) -certSerialHeader <string> **-clientCertSubject** (ENABLED | DISABLED) -certSubjectHeader <string> -clientCertHash (ENABLED | DISABLED) -certHashHeader <string> -clientCertIssuer (ENABLED | DISABLED) -certIssuerHeader <string> -sessionID (ENABLED | DISABLED) -sessionIDheader <string> -cipher (ENABLED | DISABLED) -cipherHeader <string> -clientCertNotBefore (ENABLED | DISABLED) **-certNotBeforeHeader** <string> -clientCertNotAfter (ENABLED | DISABLED) -certNotAfterHeader <string> -OWASupport (ENABLED | DISABLED)
- show ssl action [<name>]

Example

```
> add ssl action Action-SSL-ClientCert -clientCert ENABLED -certHeader "X-Client-Cert"
 Done
> show ssl action Action-SSL-ClientCert
1)      Name: Action-SSL-ClientCert
         Data Insertion Action:
        Cert Header: ENABLED            Cert Tag: X-Client-Cert
 Done
```

To configure an SSL action by using the configuration utility

Navigate to Traffic Management > SSL > Policies and, on the Actions tab, click Add.

Configuring SSL Policies

Policies on the NetScaler help identify specific connections that you want to process. The processing is based on the actions that are configured for that particular policy. Once you create the policy and configure an action for it, you must either bind it to a virtual server on the NetScaler, so that it applies only to traffic flowing through that virtual server, or bind it globally, so that it applies to all traffic flowing through any virtual server configured on the NetScaler.

The NetScaler SSL feature supports both classic policies and default syntax policies . For a complete description of classic and default syntax expressions, how they work, and how to configure them manually, see .

Note: Users who are not experienced in configuring policies at the NetScaler command line will usually find using the configuration utility considerably easier.

Configuring an SSL Default Syntax Policy

An SSL default syntax policy defines a control or a data action to be performed on requests. SSL policies can therefore be categorized as control policies and data policies:

- **Control policy**. A control policy uses a control action, such as forcing client authentication.
  Note: In release 10.5 or later, deny SSL renegotiation (denySSLReneg) is set, by default, to ALL. However, control policies, such as CLIENTAUTH, trigger a renegotiation handshake. If you use such policies, you must set denySSLReneg to NO.
- **Data policy**. A data policy uses a data action, such as inserting some data into the request.

The essential components of a policy are an expression and an action. The expression identifies the requests on which the action is to be performed. SSL policies use the default expression syntax or the classic expression syntax. For information about expressions and how to configure them, see .

You can configure a default syntax policy with a built-in action or a user-defined action. You can configure a policy with a built-in action without creating a separate action. However, to configure a policy with a user-defined action, first configure the action and then configure the policy.

You can specify an additional action, called an UNDEF action, to be performed in the event that applying the expression to a request has an undefined result.

To configure an SSL default syntax policy by using the command line interface

At the command prompt, type:

```
add ssl policy <name> -rule <expression> -Action <string> [-undefAction <string>] [-comment <string>]
```

To configure an SSL default syntax policy by using the configuration utility

Navigate to Traffic Management > SSL > Policies and, on the Polices tab, click Add.

Configuring Built-in Actions for SSL Default Syntax Policies

Unless you need only the built-in actions in your policies, you have to create the actions before creating the policies, so that you can specify the actions when you create the policies. The built-in actions are of two types, control actions and data actions. You use control actions in control policies, and data actions in data policies.
The built-in control actions are:

- CLIENTAUTHâ€”Perform client certificate authentication.
- NOCLIENTAUTHâ€”Do not perform client certificate authentication.

The built-in data actions are:

- RESETâ€”Close the connection by sending a RST packet to the client.
- DROPâ€—Drop all packets from the client. The connection remains open until the client closes it.
- NOOPâ€—Forward the packet without performing any operation on it.

You can create user-defined data actions. For example, if you enable client authentication, you can create an SSL action to insert client-certificate data into the request header before forwarding the request to the web server. For more information about user-defined actions, see Configuring User-Defined SSL Actions.

If a policy evaluation results in an undefined state, an UNDEF action is performed. For either a data policy or a control policy, you can specify RESET, DROP, or NOOP as the UNDEF action. For a control policy, you also have the option of specifying CLIENTAUTH or NOCLIENTAUTH.

Examples of built-in actions in a policy

In the following example, if the client sends a cipher other than an EXPORT category cipher, the NetScaler appliance requests client authentication. The client has to provide a valid certificate for a successful transaction.

```
add ssl policy pol1 -rule CLIENT.SSL.CIPHER_EXPORTABLE.NOT -reqAction CLIENTAUTH
```

The following examples assume that client authentication is enabled.

If the version in the certificate provided by the user matches the version in the policy, no action is taken and the packet is forwarded:

```
add ssl policy pol1 -rule CLIENT.SSL.CLIENT_CERT.VERSION.EQ(2) -reqAction NOOP
```

If the version in the certificate provided by the user matches the version in the policy, the connection is dropped:

```
add ssl policy pol1 -rule CLIENT.SSL.CLIENT_CERT.VERSION.EQ(2) -reqAction DROP
```

If the version in the certificate provided by the user matches the version in the policy, the connection is reset:

```
add ssl policy pol1 -rule CLIENT.SSL.CLIENT_CERT.VERSION.EQ(2) -reqAction RESET
```

Configuring SSL Policy Labels

Policy labels are holders for policies. A policy label helps in managing a group of policies, called a policy bank, which can be invoked from another policy. SSL policy labels can be control labels or data labels, depending on the type of policies that are included in the policy label. You can add only data policies in a data policy label and only control policies in a control policy label. To create the policy bank, you bind policies to the label and specify the order of evaluation of each policy relative to others in the bank of policies for the policy label. At the NetScaler command line, you enter two commands to create a policy label and bind policies to the policy label. In the configuration utility, you select options from a dialog box.

To create an SSL policy label and bind policies to the label by using the command line interface

At the command prompt, type:

- add ssl policylabel <labelName> -type (CONTROL | DATA)
- bind ssl policylabel <labelName> <policyName> <priority> [<gotoPriorityExpression>] [-invoke (<labelType> <labelName>)]

Example

```
add ssl policylabel cpl1 -type CONTROL
add ssl policylabel dpl1 -type DATA
bind ssl policylabel cpl1 -policyName ctrlpol -priority 1
bind ssl policylabel dpl1 -policyName datapol -priority 1
```

To configure an SSL policy label and bind policies to the label by using the configuration utility

Navigate to Traffic Management > SSL > Policy Labels, and configure an SSL policy label.

Configuring Per-Directory Client Authentication

If you create an action specifying client-side authentication on a per-directory basis, a client identified by a policy associated with the action is not authenticated as part of the initial SSL handshake. Instead, authentication is carried out every time the client wants to access a specific directory on the web server.

For example, if you have multiple divisions in the company, where each division has a folder in which all its files are stored, and you want to know the identity of each client that tries to access files from a particular directory, such as the finance directory, you can enable per-directory client authentication for that directory.

To enable per-directory client authentication, first configure client authentication as an SSL action, and then create a policy that identifies the directory that you want to monitor. When you create the policy, specify your client-authentication action as the action associated with the policy. Then, bind the policy to the SSL virtual server that will receive the SSL traffic.

To create an SSL action and a policy to enable client authentication by using the command line interface

At the command prompt, type the following commands to create an SSL action to enable to client authentication and verify the configuration:

- add ssl action <name> [-clientAuth (DOCLIENTAUTH | NOCLIENTAUTH)]
- show ssl action [<name>]
- add ssl policy <name> -rule <expression> [-action <string>] [-undefAction <string>] [-comment <string>]
- show ssl policy [<name>]

Example

```
> add ssl action ssl-action-1 -clientAuth DOCLIENTAUTH
 Done
> show ssl action ssl-action-1
1)      Name: ssl-action-1
        Client Authentication Action: DOCLIENTAUTH
                                                                Hits: 0
                                                         Undef Hits: 0
                                                                Action Reference Count: 1
 Done
> add ssl policy ssl-pol-1 -rule 'REQ.HTTP.METHOD==GET' -reqaction ssl-action-1
> sh ssl policy ssl-pol-1
                                                                Name: ssl-pol-1
                                                                Rule: REQ.HTTP.METHOD == GET
                                                                Action: ssl-action-1
                                                                UndefAction: Use Global
                                                         Hits: 0
                                                         Undef Hits: 0
 Done
```

To create an SSL action to enable client authentication by using the configuration utility

1. Navigate to Traffic Management > SSL > Policies and, on the Actions tab, click Add.
2. In the Client Authentication list, select Enabled.

To create and bind an SSL policy to enable client authentication by using the configuration utility

1. Navigate to Traffic Management > SSL and, on the Polices tab, click Add.
2. Navigate to Traffic Management > Load Balancing > Virtual Servers and open an SSL virtual server.

   In Advanced Settings, select SSL Policy, and bind the policy to the virtual server.

Configuring Support for Outlook Web Access

If your SSL configuration is offloading SSL transactions from an Outlook Web Access (OWA) server, you must insert a special header field, FRONT-END-HTTPS: ON, in all HTTP requests directed to the OWA servers. This is required for the OWA servers to generate URL links as https:// instead of http://.

When you enable support for OWA on the NetScaler, the header is automatically inserted into the specified HTTP traffic, and you do not need to configure a specific header insertion. Use SSL policies to identify all traffic directed to the OWA server.

Note: You can enable Outlook Web Access support for HTTP-based SSL virtual servers and services only. You cannot apply it to TCP-based SSL virtual servers and services.

To enable OWA support, first configure OWA support as an SSL action, and then create a policy that identifies the virtual servers or services for which you want to enable OWA support. When you create the policy, specify your OWA support action as the action associated with the policy. Then, bind the policy to the SSL virtual server that will receive the SSL traffic.

To create an SSL action and a policy to enable OWA support by using the command line interface

At the command prompt, type the following commands to create an SSL action to enable OWA support and verify the configuration:

- add ssl action <name> -OWASupport (ENABLED | DISABLED)
- show ssl action [<name>]
- add ssl policy <name> -rule <expression> [-action <string>] [-undefAction <string>] [-comment <string>]
- show ssl policy [<name>]

Example

```
> add ssl action ssl-action-2 -OWASupport ENABLED
 Done
> show ssl action ssl-action-2
1)      Name: ssl-action-2
        Type: Data Insertion
        OWA Support: ENABLED
                                                                 Hits: 0
                                                          Undef Hits: 0
                                                                 Action Reference Count: 1
 Done
> add ssl policy ssl-pol -rule 'REQ.HTTP.METHOD == GET' -reqaction ssl-action-2
Done
> sh ssl policy ssl-pol
                                                         Name: ssl-pol
                                                         Rule: REQ.HTTP.METHOD == GET
                                                         Action: ssl-action-2
                                                         UndefAction: Use Global
                                                         Hits: 0
                                                         Undef Hits: 0
Done
```

To create an SSL action to enable OWA support by using the configuration utility

1. Navigate to Traffic Management > SSL > Policies and, on the Actions tab, click Add.
2. In the Outlook Web Access list, select Enabled.
   Note: Outlook Web Access support is applicable only for SSL virtual server based configurations and transparent SSL service based configurations and not for SSL configurations with back-end encryption.

To create and bind an SSL policy to enable OWA support by using the configuration utility

Navigate to Traffic Management > SSL > Policies, and add a policy.

In the Action list, select the action that you created earlier. Specify an undefined action, and an expression.

Configuring SSL-Based Header Insertion

Because the NetScaler appliance offloads all SSL-related processing from the servers, the servers receive only HTTP traffic. In some circumstances, the server needs certain SSL information. For example, security audits of recent SSL transactions require the client subject name (contained in an X509 certificate) to be logged on the server.

Such data can be sent to the server by inserting it into the HTTP header as a name-value pair. You can insert the entire client certificate, if required, or only the specific fields from the certificate, such as the subject, serial number, issuer, certificate hash, SSL session ID, cipher suite, or the not-before or not-after date used to determine certificate validity.

You can enable SSL-based insertion for HTTP-based SSL virtual servers and services only. You cannot apply it to TCP-based SSL virtual servers and services. Also, client authentication must be enabled on the SSL virtual server, because the inserted values are taken from the client certificate that is presented to the virtual server for authentication.

To configure SSL-based header insertion, first create an SSL action for each specific set of information to be inserted, and then create policies that identify the connections for which you want to insert the information. As you create each policy, specify the action that you want associated with the policy. Then, bind the policies to the SSL virtual servers that will receive the SSL traffic.

The following example uses default syntax policies. In the following example, a control policy (ctrlpol) is created to perform client authentication if a request is received for the URL /testsite/file5.html. A data policy (datapol) is created to perform an action (act1) if client authentication is successful, and an SSL action (act1) is added to insert the certificate details and issuer's name in the request before forwarding the request. For other URLs, client authentication is disabled. The policies are then bound to an SSL virtual server (ssl_vserver) that receives the SSL traffic.

Command-line example of configuring SSL-based header insertion

Example

```
> add ssl action act1 -clientCert ENABLED -certHeader mycert -clientcertissuer ENABLED -cert
> add ssl policy datapol -rule HTTP.REQ.URL.EQ(\"/testsite/file5.html\") -action act1
> add ssl policy ctrlpol -rule HTTP.REQ.URL.EQ(\"/testsite/file5.html\") -action CLIENTAUTH
> bind ssl vserver ssl_vserver -policyName ctrlpol -priority 1
> bind ssl vserver ssl_vserver -policyName datapol -priority 1
 Done
```

To configure SSL-based header insertion by using the configuration utility

1. Navigate to Traffic Management > SSL > Policies.
2. In the details pane, on the Actions tab, click Add.
3. In the Create SSL Action dialog box, set the following parameters:
    - Name*
    - Client Certificate
    - Certificate Tag
    - Client Certificate Issuer
    - Issuer Tag

    * A required parameter
4. Click Create, and then click Close.
5. On the tab, click Add to create a control policy.
6. In the Create SSL Policy dialog box, set the following parameters:
    - Name*
    - Expression
    - Request Action

    * A required parameter
7. Click Create, and then click Close.
8. Create a data policy by repeating steps 5 through 7.
9. In the navigation pane, expand SSL Offload, and then click Virtual Servers.
10. In the details pane, from the list of virtual servers, select the virtual server to which you want to bind the SSL policies, and then click Open.
11. In the Configure Virtual Server (SSL Offload) dialog box, click SSL Settings, and then click SSL Policies.
12. In the Bind/Unbind SSL Policies dialog box, click Insert Policy. Under Policy Name, select the policy that you created in steps 5 through 7.
13. Click OK, and then click Close. A message appears in the status bar, stating that the policy has been bound successfully.

14. Repeat steps 12 and 13 and select the policy that you created in step 8.

Binding SSL Policies to a Virtual Server

The SSL policies that are configured on the NetScaler appliance need to be bound to a virtual server that intercepts traffic directed to the virtual server. If the incoming data matches any of the rules configured in the SSL policy, the policy is triggered and the action associated with it is carried out.

You can also bind SSL policies globally or to custom bind points on the NetScaler appliance. For more information about binding policies on the appliance, see .

To bind an SSL policy to a virtual server by using the command line interface

At the command prompt, type the following command to bind an SSL policy to a virtual server and verify the configuration:

- bind ssl vserver <vServerName> -policyName <string> [-priority <positive_integer>]
- show ssl vserver <vServerName>

Example

```
> bind ssl vserver vs-server -policyName ssl-policy-1 -priority 10
 Done
> show ssl vserver vs-server

        Advanced SSL configuration for VServer vs-server:
        DH: DISABLED
        Ephemeral RSA: ENABLED          Refresh Count: 1000
        Session Reuse: ENABLED          Timeout: 120 seconds
        Cipher Redirect: DISABLED
        SSLv2 Redirect: DISABLED
        ClearText Port: 80
        Client Auth: DISABLED
        SSL Redirect: ENABLED
        SSL-REDIRECT Port Rewrite: ENABLED
        Non FIPS Ciphers: DISABLED
        SSLv2: DISABLED SSLv3: ENABLED  TLSv1: ENABLED

1)      Policy Name: ssl-policy-1        Priority: 10

1)      Cipher Name: DEFAULT
        Description: Predefined Cipher Alias
 Done
```

To bind an SSL policy to a virtual server by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers, and open an SSL virtual server.
2. In Advanced Settings, select SSL Policy, Click in the SSL Policy section to bind to the virtual server.

Binding SSL Policies Globally

Globally bound policies are evaluated after all policies bound to services, virtual servers, or other NetScaler bind points are evaluated.

To globally bind an SSL policy by using the command line interface

At the command prompt, type the following command to bind a global SSL policy and verify the configuration:

- bind ssl global - policyName <string> [- priority <positive_integer>]
- show ssl global

Example

```
> bind ssl global -policyName Policy-SSL-2 -priority 90
Done
> sh ssl global
1) Name: Policy-SSL-2 Priority: 90
2) Name: Policy-SSL-1 Priority: 100
Done
```

To bind a global SSL policy by using the configuration utility

1. Navigate to Traffic Management > SSL > Policies.
2. In the details pane, click Global Bindings.
3. In the Bind/Unbind SSL Policies to Global dialog box, click Insert Policy.
4. In the Policy Name drop-down list, select a policy.
5. Optionally, drag the entry to a new position in the policy bank to automatically update the priority level.
6. Click OK. A message appears in the status bar, stating that the policy has been bound successfully.

Use Case 1: Configuring SSL Offloading with End-to-End Encryption

A simple SSL offloading setup terminates SSL traffic (HTTPS), decrypts the SSL records, and forwards the clear text (HTTP) traffic to the back-end web servers. However, the clear text traffic is vulnerable to being spoofed, read, stolen, or compromised by individuals who succeed in gaining access to the back-end network devices or web servers.

You can, therefore, configure SSL offloading with end-to-end security by re-encrypting the clear text data and using secure SSL sessions to communicate with the back-end Web servers.

Additionally, you can configure the back-end SSL transactions so that the NetScaler appliance uses SSL session multiplexing to reuse existing SSL sessions with the back-end web servers, thus avoiding CPU-intensive key exchange (full handshake) operations. This reduces the overall number of SSL sessions on the server, and therefore accelerates the SSL transaction while maintaining end-to-end security.

To configure SSL Offloading with end-to-end encryption, add SSL based services that represent secure servers with which the NetScaler appliance will carry out end-to-end encryption. Then create an SSL based virtual server, and create and bind a valid certificate-key pair to the virtual server. Bind the SSL services to the virtual server to complete the configuration.

For details on adding SSL based services, see Configuring Services.

For details on adding an SSL virtual server, see Configuring an SSL Based Virtual Server.

For details on creating a certificate-key pair, see Adding a Certificate-Key Pair.

For details on binding a certificate-key pair to a virtual server, see Binding the Certificate Key Pair to the SSL Based Virtual Server.

For details on binding services to a virtual server, see Binding Services to the SSL Based Virtual Server.

Example

Create two SSL based services, Service-SSL-1 and Service-SSL-2, with IP addresses 10.102.20.30 and 10.102.20.31 and both using port 443.

Then create an SSL based virtual server, Vserver-SSL-2 with an IP address of 10.102.10.20.

Next, create a certificate-key pair, CertKey-1 and bind it to the virtual server.

Bind the SSL services to the virtual server to complete the configuration.

Table 1. Entities in the SSL Offloading with End-to-End Encryption Example

Entity	Name	Value
SSL Service	Service-SSL-1	10.102.20.30
Â	Service-SSL-2	10.102.20.31
SSL Based Virtual Server	Vserver-SSL-2	10.102.10.20
Certificate - Key Pair	Certkey-1	Â

Use Case 2: Configuring Transparent SSL Acceleration

Note: You need to enable L2 mode on the NetScaler appliance for transparent SSL acceleration to work.

Transparent SSL acceleration is useful for running multiple applications on a secure server with the same public IP, and also for SSL acceleration without using an additional public IP.

In a transparent SSL acceleration setup, the NetScaler appliance is transparent to the client, because the IP address at which the appliance receives requests is the same as the Web server's IP address.

The NetScaler offloads SSL traffic processing from the Web server and sends either clear text or encrypted traffic (depending on the configuration) to the web server. All other traffic is transparent to the NetScaler and is bridged to the Web server. Therefore, other applications running on the server are unaffected.

There are three modes of transparent SSL acceleration available on the NetScaler:

- Service-based transparent access, where the service type can be SSL or SSL_TCP.
- Virtual server-based transparent access with a wildcard IP address (*:443).
- SSL VIP-based transparent access with end-to-end encryption.

Note: An SSL_TCP service is used for non-HTTPS services (for example SMTPS and IMAPS).

Service-based Transparent SSL Acceleration

To enable transparent SSL acceleration using the SSL service mode, configure an SSL or an SSL_TCP service with the IP address of the actual back-end Web server. Instead of a virtual server intercepting SSL traffic and passing it on to the service, the traffic is now directly passed on to the service, which decrypts the SSL traffic and sends clear text data to the back-end server.

The service-based mode allows you to configure individual services with a different certificate, or with a different clear text port. Also, you can also select individual services for SSL acceleration.

You can apply service-based transparent SSL acceleration to data that uses different protocols, by setting the clear text port of the SSL service to the port on which the data transfer between the SSL service and the back-end server occurs.

To configure service-based transparent SSL acceleration, first enable both the SSL and the load balancing features. Then create an SSL based service and configure its clear text port. After the service is created, create and bind a certificate-key pair to this service.

For details on configuring the clear text port for an SSL based service, see "Configuring Advanced SSL Settings."

For details on creating a certificate-key pair and binding a certificate-key pair to a service, see "Adding a Certificate-Key Pair."

Example

Enable SSL offloading and load balancing.

Create an SSL based service, Service-SSL-1 with the IP address 10.102.20.30 using port 443 and configure its clear text port.

Next, create a certificate-key pair, CertKey-1 and bind it to the SSL service.

Table 1. Entities in the Service-based Transparent SSL Acceleration

Entity	Name	Value
SSL Service	Service-SSL-1	102.20.30
Certificate - Key Pair	Certkey-1	

Virtual Server-based Acceleration with a Wildcard IP Address (*:443)

You can use an SSL virtual server in the wildcard IP address mode if when you want to enable SSL acceleration for multiple servers that host the secure content of a Web site. In this mode, a single-digital certificate is enough for the entire secure Web site, instead of one certificate per virtual server. This results in significant cost savings on SSL certificates and renewals. The wildcard IP address mode also enables centralized certificate management.

To configure global transparent SSL acceleration on the NetScaler appliance, create a *:443 virtual server, which is a virtual server that accepts any IP address associated with port 443. Then, bind a valid certificate to this virtual server, and also bind all services to which the virtual server is to transfer. Such a virtual server can use the SSL protocol for HTTP-based data or the SSL_TCP protocol for non-HTTP-based data.

To configure virtual server-based acceleration with a wildcard IP address

1. Enable SSL, as described in "Enabling SSL Processing."
2. Enable load balancing, as described in "Load Balancing."
3. Add an SSL based virtual server (see "Configuring an SSL-Based Virtual Server" for the basic settings), and set the clearTextPort parameter (described in "Configuring Advanced SSL Settings)."
4. Add a certificate-key pair, as described in "Adding a Certificate-Key Pair."
   Note: The wildcard server will automatically learn the servers configured on the NetScaler, so you do not need to configure services for a wildcard virtual server.

Example

After enabling SSL offloading and load balancing, create an SSL based wildcard virtual server with IP address set to * and port number 443, and configure its clear text port (optional).

If you specify the clear text port, decrypted data will be sent to the backend server on that particular port. Otherwise, encrypted data will be sent to port 443.

Next, create an SSL certificate key pair, CertKey-1 and bind it to the SSL virtual server.

Table 2. Entities in the Virtual Server-based Acceleration with a Wildcard IP Address Example

Entity	Name	IP Address	Port
SSL Based Virtual Server	Vserver-SSL-Wildcard	*	443
Certificate - Key Pair	Certkey-1	Â	Â

SSL VIP-based Transparent Access with End-To-End Encryption

You can use an SSL virtual server for transparent access with end-to-end encryption if you have no clear text port specified. In such a configuration, the NetScaler terminates and offloads all SSL processing, initiates a secure SSL session, and sends the encrypted data, instead of clear text data, to the web servers on the port that is configured on the wildcard virtual server.

Note: In this case, the SSL acceleration feature runs at the back-end, using the default configuration, with all 34 ciphers available.

To configure SSL VIP based transparent access with end-to-end encryption, Follow instructions for Configuring a Virtual Server-based Acceleration with a Wildcard IP Address (*:443), but do not configure a clear text port on the virtual server.

Use Case 3: Configuring SSL Acceleration with HTTP on the Front End and SSL on the Back End

In certain deployments, you might be concerned about network vulnerabilities between the NetScaler appliance and the backend servers, or you might need complete end-to-end security and interaction with certain devices that can communicate only in clear text (for example, caching devices).

In such cases, you can set up an HTTP virtual server that receives data from clients that connect to it at the front end and hands the data off to a secure service, which securely transfers the data to the web server.

To implement this type of configuration, you configure an HTTP virtual server on the NetScaler and bind SSL based services to the virtual server. The NetScaler receives HTTP requests from the client on the configured HTTP virtual server, encrypts the data, and sends the encrypted data to the web servers in a secure SSL session.

To configure SSL acceleration with HTTP on the front-end and SSL on the back-end, first enable the load balancing and SSL features on the NetScaler. Then, add SSL based services that represent secure servers to which the NetScaler appliance will send encrypted data. Finally, add an HTTP based virtual server and bind the SSL services to this virtual server.

Example

Enable load balancing and SSL acceleration on the NetScaler.

After enabling load balancing and SSL acceleration, create two SSL based services, Service-SSL-1 and Service-SSL-2, with IP addresses 10.102.20.30 and 10.102.20.31, and both using port 443.

Then create an HTTP based virtual server, Vserver-HTTP-1, with an IP address of 10.102.10.20.

Bind the SSL services to the virtual server to complete the configuration.

Table 1. Entities in the SSL Acceleration with HTTP on the Front End and SSL on the Back End Example

Entity	Name	Value
SSL Service	Service-SSL-1	10.102.20.30
Â	Service-SSL-2	10.102.20.31
HTTP Based Virtual Server	Vserver-HTTP-1	10.102.10.20

Use Case 4: SSL Offloading with Other TCP Protocols

In addition to the secure HTTP (HTTPS) protocol, NetScaler appliances support SSL acceleration for other TCP-based secure protocols. However, only simple requests and response-based TCP application protocols are supported. Applications such as FTPS, that insert the server's IP address and port information in their payloads, are not currently supported.

Note: The STARTTLS feature for SMTP is currently not supported.

The NetScaler supports SSL acceleration for Other TCP protocols with and without end-to-end encryption.

To configure SSL offloading with Other TCP protocols, create a virtual server of type SSL_TCP, bind a certificate-key pair and TCP based services to the virtual server, and configure SSL actions and policies based on the type of traffic expected and the acceleration to be provided.

Follow the instructions in Configuring SSL Offloading, but create an SSL_TCP virtual server instead of an SSL virtual server, and configure TCP services instead of HTTP services.

SSL_TCP Based Offloading with End-to-End Encryption

To configure SSL_TCP-based offloading with end-to-end encryption, both the virtual server that intercepts secure traffic and the services that it forwards the traffic to must be of type SSL_TCP.

Configure SSL_TCP-based offloading as described in Configuring SSL Offloading with End-to-End Encryption, but create an SSL_TCP virtual server instead of an SSL virtual server.

Backend Encryption for TCP Based Data

Some deployments might require the NetScaler appliance to encrypt TCP data received as clear text and send the data securely to the back end servers.

To provide SSL acceleration with back-end encryption for clear text TCP traffic arriving from the client, create a TCP based virtual server and bind it to SSL_TCP based services.

To configure end-to-end encryption for TCP-based data, follow the procedure described in Configuring the SSL feature with HTTP on the Front-End and SSL on the Back-End, but create a TCP virtual server instead of an HTTP virtual server.

Use Case 5: Configuring SSL Bridging

An SSL bridge configured on the NetScaler appliance enables the appliance to bridge all secure traffic between the SSL client and the SSL server. The appliance does not offload or accelerate the bridged traffic, nor does it perform encryption or decryption. Only load balancing is done by the appliance. The SSL server must handle all SSL-related processing. Features such as content switching, SureConnect, and cache redirection do not work, because the traffic passing through the appliance is encrypted.

Because the appliance does not carry out any SSL processing in an SSL bridging setup, there is no need for SSL certificates.

Citrix recommends that you use this configuration only if an acceleration unit (for example, a PCI-based SSL accelerator card) is installed in the web server to handle the SSL processing overhead.

Before you configure SSL bridging, first enable SSL and load balancing on the appliance. Then, create SSL_Bridge services and bind them to an SSL_Bridge virtual server. Configure the load balancing feature to maintain server persistency for secure requests.

Example

After enabling SSL and load balancing, create two servers, s1 and s2. Create two SSL_Bridge services, sc1 and src2. Create an SSL_Bridge virtual server and bind the SSL_Bridge services to the virtual server to complete the configuration. At the command line, type:

```
enable ns feature SSL LB
add server s1 10.102.1.101
add server s2 10.102.1.102
add service src1 s1 SSL_BRIDGE 443
add service src2 s2 SSL_BRIDGE 443
add lb vserver ssl_bridge_vip SSL_BRIDGE 10.102.1.200 443
bind lb vserver ssl_bridge_vip src1
bind lb vserver ssl_bridge_vip src2
```

Use Case 6: Configuring SSL Monitoring when Client Authentication is Enabled on the Backend Service

Consider a scenario in which you need to load balance servers that require SSL client certificates to validate clients. For this deployment, you need to create an SSL service on the NetScaler appliance, add an HTTPS monitor, add a certificate-key pair, bind this certificate-key pair to the SSL service, and then bind the https monitor to this service. You can use this https monitor to perform health checks on the backend services.

To configure SSL monitoring with client certificate

1. Open an SSH connection to the appliance by using an SSH client, such as PuTTY.
2. Log on the appliance by using the administrator credentials.
3. Add an SSL service. At the command prompt, type:

   add service <name> <serverName> <serviceType> <port>
4. Add an https monitor. At the command prompt, type:

   add lb monitor <name> <type>
5. Add the certificate-key pair that is going to be used as the client cert for that SSL service. At the command prompt, type:

   add ssl certKey <certkeyName> -cert <string> -key <string>
6. Bind this certkey to the SSL service. At the command prompt, type:

   bind ssl service <serviceName> -certkeyName <string>
7. Bind the https monitor to the SSL service. At the command prompt, type:

   bind lb monitor <monitorName> <serviceName>

Now, when the appliance tries to probe the backend service on which client authentication is enabled, the backend service will request a certificate as part of the SSL handshake. When the appliance returns the certificate-key bound in step 6 above, the monitor probe will succeed.

Example

```
add service svc_k 10.102.145.30 SSL 443
add lb monitor sslmon HTTP -respCode 200 -httpRequest "GET /testsite/file5.html" -secure YE
add ssl certKey ctest -cert client_rsa_2048.pem -key client_rsa_2048.ky
bind ssl service svc_k -certkeyName ctest
bind lb monitor sslmon svc_k
> show service svc_k
      svc_k (10.102.145.30:443) - SSL
      State: UP
      Last state change was at Tue Jan 10 13:12:24 2012
      Time since last state change: 0 days, 00:09:37.890
      Server Name: 10.102.145.30
      Server ID : 0 Monitor Threshold : 0
      Max Conn: 0 Max Req: 0 Max Bandwidth: 0 kbits
      Use Source IP: NO
      Client Keepalive(CKA): NO
      Access Down Service: NO
      TCP Buffering(TCPB): NO
      HTTP Compression(CMP): NO
      Idle timeout: Client: 180 sec Server: 360 sec
      Client IP: DISABLED
      Cacheable: NO
      SC: OFF
      SP: OFF
      Down state flush: ENABLED
      Appflow logging: ENABLED

1) Monitor Name: sslmon
      State: UP Weight: 1
      Probes: 1318 Failed [Total: 738 Current: 0]
      Last response: Success - HTTP response code 200 received.
      Response Time: 0.799 millisec
   Done
>
```

```
> show ssl service svc_k
                    Advanced SSL configuration for Back-end SSL Service svc_k:
      DH: DISABLED
      Ephemeral RSA: DISABLED
      Session Reuse: ENABLED Timeout: 300 seconds
      Cipher Redirect: DISABLED
      SSLv2 Redirect: DISABLED
      Server Auth: DISABLED
      SSL Redirect: DISABLED
      Non FIPS Ciphers: DISABLED
      SNI: DISABLED
      SSLv2: DISABLED SSLv3: ENABLED TLSv1: ENABLED
1) CertKey Name: ctest Client Certificate

1) Cipher Name: ALL
   Description: Predefined Cipher Alias
Done
```

Use Case 7: Configuring a Secure Content Switching Server

An SSL-based content switching virtual server first decrypts the secure data and then redirects the data to appropriately configured servers as determined by the type of content and the configured content switching policies. The packets sent to the server have a mapped IP address as the source IP address.

The following example shows the steps to configure two address-based virtual servers to perform load balancing on the HTTP services. One virtual server, Vserver-LB-HTML, load balances the dynamic content (cgi, asp), and the other, Vserver-LB-Image, load balances the static content (gif, jpeg). The load-balancing method used is the default, LEASTCONNECTION. A content switching SSL virtual server, Vserver-CS-SSL, is then configured to perform SSL acceleration and switching of HTTPS requests on the basis of configured content switching policies.

Example

```
> enable ns feature lb cs ssl
> add lb vserver Vserver-LB-HTML http 10.1.1.2 80
> add lb vserver Vserver-LB-Image http 10.1.1.3 80
> add service s1 10.1.1.4 http 80
> add service s2 10.1.1.5 http 80
> add service s3 10.1.1.6 http 80
> add service s4 10.1.1.7 http 80
> bind lb vserver Vserver-LB-HTML s1
> bind lb vserver Vserver-LB-HTML s2
> bind lb vserver Vserver-LB-Image s3
> bind lb vserver Vserver-LB-Image s4
> add cs vserver Vserver-CS-SSL ssl 10.1.1.1 443
> add cs policy pol1 -url "*.cgi"
> add cs policy pol2 -url "*.asp"
> add cs policy pol3 -url "*.gif"
> add cs policy pol4 -url "*.jpeg"
> bind cs vserver Vserver-CS-SSL -policyName pol1 Vserver-LB-HTML
> bind cs vserver Vserver-CS-SSL -policyName pol2 Vserver-LB-HTML
> bind cs vserver Vserver-CS-SSL -policyName pol3 Vserver-LB-Image
> bind cs vserver Vserver-CS-SSL -policyName pol4 Vserver-LB-Image
> add certkey mykey -cert /nsconfig/ssl/ns-root.cert -key /nsconfig/ssl/ns-root.key
> bind certkey Vserver-CS-SSL mykey
>
> show cs vserver Vserver-CS-SSL
Â Â Â Â Â Â Â  Vserver-CS-SSL (10.1.1.1:443) - SSL Type: CONTENT
Â Â Â Â Â Â Â  State: UP
Â Â Â Â Â Â Â  Last state change was at Tue Jul 13 02:11:37 2010
Â Â Â Â Â Â Â  Time since last state change: 0 days, 00:02:12.440
Â Â Â Â Â Â Â  Client Idle Timeout: 180 sec
Â Â Â Â Â Â Â  Down state flush: ENABLED
Â Â Â Â Â Â Â  Disable Primary Vserver On Down : DISABLED
Â Â Â Â Â Â Â  State Update: DISABLED
Â Â Â Â Â Â Â  Default:Â Â Â Â Â Â Â  Content Precedence: RULE
Â Â Â Â Â Â Â  Vserver IP and Port insertion: OFF
Â Â Â Â Â Â Â  Case Sensitivity: ON
Â Â Â Â Â Â Â  Push: DISABLEDÂ  Push VServer:
Â Â Â Â Â Â Â  Push Label Rule: none
```

Ciphers Supported by the NetScaler Appliance

Your NetScaler appliance ships with a predefined set of cipher groups. To use ciphers that are not part of the DEFAULT cipher group, you have to explicitly bind them to an SSL virtual server. You can also create a user-defined cipher group to bind to the SSL virtual server. For more information about creating a user-defined cipher group, see Configuring User-Defined Cipher Groups on the NetScaler Appliance.

Table 1 lists the ciphers that are part of the DEFAULT cipher group and are therefore bound by default to an SSL virtual server.

Table 2 lists the other ciphers currently supported by NetScaler appliances.

Table 1. Ciphers That the NetScaler Appliance Supports by Default on the Front End

Cipher Suite	**Protocol**	**Key Exchange Algorithm**	**Authentication Algorithm**	**Encryption Algorithm (Key Size)**
TLS1-AES-256-CBC-SHA	SSLv3 TLSv1 TLSv1.1 TLSv1.2	RSA	RSA	AES(256)
TLS1-AES-128-CBC-SHA	SSLv3 TLSv1 TLSv1.1 TLSv1.2	RSA	RSA	AES(128)
TLS1.2-AES-256-SHA256	TLSv1.2	RSA	RSA	AES(256)
TLS1.2-AES-128-SHA256	TLSv1.2	RSA	RSA	AES(128)
TLS1.2-AES256-GCM-SHA384	TLSv1.2	RSA	RSA	AES-GCM (256)
TLS1.2-AES128-GCM-SHA256	TLSv1.2	RSA	RSA	AES-GCM (128)
TLS1-ECDHE-RSA-AES256-SHA	SSLv3 TLSv1 TLSv1.1 TLSv1.2	ECC-DHE	RSA	AES(256)
TLS1-ECDHE-RSA-AES128-SHA	SSLv3 TLSv1 TLSv1.1 TLSv1.2	ECC-DHE	RSA	AES(128)

TLS1.2-ECDHE-RSA-AES-256-SHA384	TLSv1.2	ECC-DHE	RSA	AES(256)
TLS1.2-ECDHE-RSA-AES-128-SHA256	TLSv1.2	ECC-DHE	RSA	AES(128)
TLS1.2-ECDHE-RSA-AES256-GCM-SHA384	TLSv1.2	ECC-DHE	RSA	AES-GCM (256)
TLS1.2-ECDHE-RSA-AES128-GCM-SHA256	TLSv1.2	ECC-DHE	RSA	AES-GCM (128)
TLS1.2-DHE-RSA-AES-256-SHA256	TLSv1.2	DH	RSA	AES(256)
TLS1.2-DHE-RSA-AES-128-SHA256	TLSv1.2	DH	RSA	AES(128)
TLS1.2-DHE-RSA-AES256-GCM-SHA384	TLSv1.2	DH	RSA	AES-GCM (256)
TLS1.2-DHE-RSA-AES128-GCM-SHA256	TLSv1.2	DH	RSA	AES-GCM (128)
TLS1-DHE-RSA-AES-256-CBC-SHA	SSLv3 TLSv1 TLSv1.1 TLSv1.2	DH	RSA	AES(256)
TLS1-DHE-RSA-AES-128-CBC-SHA	SSLv3 TLSv1 TLSv1.1 TLSv1.2	DH	RSA	AES(128)
TLS1-DHE-DSS-AES-256-CBC-SHA	SSLv3 TLSv1	DH	DSS	AES(256)
TLS1-DHE-DSS-AES-128-CBC-SHA	SSLv3 TLSv1	DH	DSS	AES(128)
TLS1-ECDHE-RSA-DES-CBC3-SHA	SSLv3 TLSv1 TLSv1.1 TLSv1.2	ECC-DHE	RSA	3DES(168)
SSL3-EDH-RSA-DES-CBC3-SHA	SSLv3 TLSv1 TLSv1.1 TLSv1.2	DH	RSA	3DES(168)
SSL3-EDH-DSS-DES-CBC3-SHA	SSLv3 TLSv1	DH	DSS	3DES(168)
TLS1-ECDHE-RSA-RC4-SHA	SSLv3	ECC-DHE	RSA	RC4(128)

	TLSv1 TLSv1.1 TSv1.2			
TLS1-DHE-DSS-RC4-SHA	SSLv3 TLSv1	DH	DSS	RC4(128)
SSL3-DES-CBC3-SHA	SSLv3 TLSv1 TLSv1.1 TLSv1.2	RSA	RSA	3DES(168)

Table 2. Additional Ciphers Supported by the NetScaler Appliance

Cipher Suite	**Protocol**	**Key Exchange Algorithm**	**Authentication Algorithm**	**Encryption Algorithm (Key Size)**	**Message Authentic Code (MA Algorithm**
SSL3-DES-CBC-SHA	SSLv3 TLSv1 TLv1.1	RSA	RSA	DES(56)	SHA1
TLS1-EXP1024-RC4-SHA	TLSv1	RSA (1024)	RSA	RC4(56)	SHA1 Exp
SSL3-EXP-RC4-MD5	SSLv3 TLSv1	RSA(512)	RSA	RC4(40)	MD5 Expo
SSL3-EXP-DES-CBC-SHA	SSLv3 TLSv1	RSA(512)	RSA	DES(40)	SHA1 Exp
SSL3-EXP-RC2-CBC-MD5	SSLv3 TLSv1	RSA(512)	RSA	RC2(40)	MD5 Expo
SSL2-RC4-MD5	SSLv2	RSA	RSA	RC4(128)	MD5
SSL2-DES-CBC3-MD5	SSLv2	RSA	RSA	3DES(168)	MD5
SSL2-RC2-CBC-MD5	SSLv2	RSA	RSA	RC2(128)	MD5
SSL2-DES-CBC-MD5	SSLv2	RSA	RSA	DES(56)	MD5
SSL2-RC4-64-MD5	SSLv2	RSA	RSA	RC4(64)	MD5
SSL2-EXP-RC4-MD5	SSLv2	RSA(512)	RSA	RC4(40)	MD5 Expo
SSL3-EDH-DSS-DES-CBC-SHA	SSLv3	DH	DSS	DES(56)	SHA1

	TLSv1				
TLS1-EXP1024-DHE-DSS-DES-CBC- SHA	TLSv1	DH(1024)	DSS	DES(56)	SHA1 Exp
TLS1-EXP1024-DHE-DSS-RC4- SHA	TLSv1	DH(1024)	DSS	RC4(56)	SHA1 Exp
SSL3-EXP-EDH-DSS-DES-CBC-SHA	SSLv3 TLSv1	DH(512)	DSS	DES(40)	SHA1 Exp
SSL3-EDH-RSA-DES-CBC-SHA	SSLv3 TLSv1 TLSv1.1	DH	RSA	DES(56)	SHA1
SSL3-EXP-EDH-RSA-DES-CBC-SHA	SSLv3 TLSv1	DH(512)	RSA	DES(40)	DES(40)
TLS1-EXP1024-RC4-MD5	TLSv1	RSA (1024)	RSA	RC4(56)	MD5 Expo
TLS1-EXP1024-RC2-CBC-MD5	TLSv1	RSA (1024)	RSA	RC2(56)	MD5 Expo
SSL2-EXP-RC2-CBC-MD5	SSLv2	RSA(512)	RSA	RC2(40)	MD5 Expo
SSL3-ADH-RC4-MD5	SSLv3 TLSv1 TLSv1.1	DH	None	RC4(128)	MD5
SSL3-ADH-DES-CBC-SHA	SSLv3 TLSv1 TLSv1.1	DH	None	DES(56)	SHA1
SSL3-ADH-DES-CBC3-SHA	SSLv3 TLSv1 TLSv1.1	DH	None	3DES(168)	SHA1
TLS1-ADH-AES-128-CBC-SHA	SSLv3 TLSv1 TLSv1.1	DH	None	AES(128)	SHA1
TLS1-ADH-AES-256-CBC-SHA	SSLv3 TLSv1 TLSv1.1	DH	None	AES(256)	SHA1
SSL3-EXP-ADH-RC4-MD5	SSLv3 TLSv1	DH(512)	None	RC4(40)	MD5 Expo
SSL3-EXP-ADH-DES-CBC-SHA	SSLv3	DH(512)	None	DES(40)	SHA1 Exp

	TLSv1				
SSL3-RC4-SHA	SSLv3 TLSv1 TLSv1.1 TLSv1.2	RSA	RSA	RC4(128)	SHA1
SSL3-RC4-MD5	SSLv3 TLSv1 TLSv1.1 TLSv1.2	RSA	RSA	RC4(128)	MD5

Note

1. The following curves are supported for ECDHE key exchange algorithms:
   - ECDHE 521 curve
   - ECDHE 384 curve
   - ECDHE 256 curve
   - ECDHE 224 curve
     For more information about the ECDHE ciphers supported on a NetScaler appliance, see Configuring ECDHE Ciphers.
2. The NetScaler VPX appliance supports only the SHA256 + RSA (Hex code 0x0401) signature hash algorithm in the certificate request message. If you use a different signature algorithm to sign the certificate verify message, the TLSv1.2 handshake fails during client authentication with the error message "Unsupported Certificate."
3. AES-GCM/SHA2 ciphers are supported on both the front end and back end SSL entities on an MPX appliance, and only on the front end SSL entities on VPX and SDX appliances.

Cipher/Protocol Support Matrix on the NetScaler Appliance

From release 10.5 build 56.22, NetScaler MPX appliances support full hardware optimization for all ciphers. In earlier releases, part of ECDHE/DHE cipher optimization was done in software.

Note: Hardware optimization is not supported for ciphers that are specific to the NetScaler VPX appliance. On the SDX platform, if you do not assign an SSL chip to an instance, optimization is done by software.

The following tables list the support for different ciphers on SSL entities, such as virtual server, front-end, back-end, and internal services. Use the 'show hardware' command to identify whether your appliance has N3 chips.

On an SDX appliance, if an SSL chip is assigned to a VPX instance, the cipher support of an MPX appliance applies. Otherwise, the normal cipher support of a VPX instance applies.

Example

```
> sh hardware

Platform: NSMPX-22000 16*CPU+24*IX+12*E1K+2*E1K+4*CVM N3 2200100

Manufactured on: 8/19/2013

CPU: 2900MHZ

Host Id: 1006665862

Serial no: ENUK6298FT

Encoded serial no: ENUK6298FT

Done
```

Note

1. TLS-Fallback_SCSV cipher suite is supported on all appliances from release 10.5 build 57.x
2. HTTP Strict Transport Security (HSTS) support is policy-based.
3. All SHA-2 signed-certificates (SHA256, SHA384, SHA512) are supported only on the front end and only SHA256 signed-certificates are supported on the back end of all appliances.
4. From release 11.1, the following ciphers are supported on the MPX 9700/10500/12500/15500 FIPS appliances:
   - TLS1.2-ECDHE-RSA-AES-256-SHA384 (only front end)
   - TLS1.2-ECDHE-RSA-AES-128-SHA256 (front end and back end)
5. ECDSA is supported only on the front end of MPX appliances that contain N3 chips.

Table1: Support on Virtual Server/Frontend Service/Internal Service

	MPX/SDX (N2)	MPX/SDX (N3)	VPX	MPX 9700* FIPS with firmware 2.2	MPX 1 FIPS
TLS 1.1/1.2	10.0	10.0	10.5-57. x	10.5 58.1108.e	10.5-5 e
ECDHE/DHE (Example TLS1-ECDHE-RSA-AES128-SHA)	10.5-53.x	10.1-124. x	10.5	11.1/ 10.5-59.1306.e	Not su
AES-GCM (Example TLS1.2-AES128-GCM-SHA256)	10.5-53.x	10.5-53.x	11.0-66. x / 11.1	See note	Not su

SHA-2 Ciphers (Example TLS1.2-AES-128-SHA256)	10.5-53.x	10.5-53.x	11.0-66. x / 11.1	See note	Not su
ECDSA (Example TLS1-ECDHE-ECDSA-AES256-SHA)	Not supported	11.1 (only MPX)	Not supported	Not applicable	Not su

Table 2: Support on Backend Services

	MPX/SDX (N2)	MPX/SDX (N3)	VPX	MPX 9700* FIPS with firmware 2.2
TLS 1.1/1.2	10.5-59.x/11.0-50. x	10.5-59.x /11.0-50. x	11.0-66.x	10.5 58.1108 e
ECDHE/DHE (Example TLS1-ECDHE-RSA-AES128-SHA)	10.5-58.x/11.0-50. x	10.5-58.x/11.0-50. x	Not supported	11.1/10. 5 59.1306 e
AES-GCM (Example TLS1.2-AES128-GCM-SHA256)	11.1	11.1	Not supported	See note
SHA-2 (Example TLS1.2-AES-128-SHA256)	11.1	11.1	Not supported	See note
ECDSA (Example TLS1-ECDHE-ECDSA-AES256-SHA)	Not supported	Not supported	Not supported	Not appl

* MPX 9700/10500/12500/15500

** MPX 14030/14060/14080

Server Certificate Support Matrix on the NetScaler Appliance

The NetScaler appliance supports the following certificates from release 9.3.

Table 1: Support on Frontend (FE) and Backend (BE) Service

	MPX/SDX (N2 CHIPS)		MPX/SDX (N3 CHIPS)		VPX		MPX 9700/10500/12500/15500 FIPS with FW 2.2		MPX/SDX 14030/14060/1 FIPS	
	FE	BE	FE	BE	FE	BE	FE	BE	FE	B
MD5	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SHA1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SHA224	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SHA256	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SHA384	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SHA512	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
RSA Key	Up to 4096 bits	Up to 4096 bits	Up to 4096 bits	Up to 4096 bits	Up to 4096 bits	Up to 2048 bits	Up to 2048 bits	Up to 2048 bits	2048 and 3072 bits	2 a 3 b
DH Key	Up to 2048 bits	Up to 2048 bits	Up to 2048 bits	Up to 2048 bits	Up to 2048 bits	Up to 2048 bits	N	N	N	N

Note

The NetScaler appliance supports the following "signature algorithms" extensions in the back end client hello message:

- RSA-MD5
- RSA-SHA1
- RSA-SHA256

Because SHA 384 and SHA 512 signature algortihms extensions are not supported by the NetScaler appliance, some servers, such as Windows IIS servers, reset the connection.

FIPS

The Federal Information Processing Standard (FIPS), issued by the US National Institute of Standards and Technologies, specifies the security requirements for a cryptographic module used in a security system. The NetScaler FIPS appliance complies with the second version of this standard, FIPS-140-2.

Note: Henceforth, all references to FIPS imply FIPS-140-2.

The FIPS appliance is equipped with a tamper-proof (tamper-evident) cryptographic moduleâ€"and a Cavium CN1620-NFBE3-2.0-G on the MPX 9700/10500/12500/15500 FIPS appliancesâ€"designed to comply with the FIPS 140-2 Level-2 specifications. The Critical Security Parameters (CSPs), primarily the server's private-key, are securely stored and generated inside the cryptographic module, also referred to as the Hardware Security Module (HSM). The CSPs are never accessed outside the boundaries of the HSM. Only the superuser (nsroot) can perform operations on the keys stored inside the HSM.

The following table summarizes the differences between standard NetScaler and NetScaler FIPS appliances.

Setting	NetScaler appliance	NetScaler FIPS appliance
Key storage	On the hard disk	On the FIPS card
Cipher support	All ciphers	FIPS approved ciphers
Accessing keys	From the hard disk	Not accessible

Configuring a FIPS appliance involves configuring the HSM immediately after completing the generic configuration process. You then create or import a FIPS key. After creating a FIPS key, you should export it for backup. You might also need to export a FIPS key so that you can import it to another appliance. For example, configuring FIPS appliances in a high availability (HA) setup requires transferring the FIPS key from the primary node to the secondary node immediately after completing the standard HA setup.

You can upgrade the firmware version on the FIPS card from version 4.6.0 to 4.6.1, and you can reset an HSM that has been locked to prevent unauthorized logon. Only FIPS approved ciphers are supported on a NetScaler FIPS appliance.

This section includes the following details:

- Configuring the HSM
- Creating and Transferring FIPS Keys
- Configuring FIPS Appliances in a High Availability Setup
- Resetting a Locked HSM
- FIPS Approved Algorithms and Ciphers

Configuring the HSM

Before you can configure the HSM of your NetScaler FIPS appliance, you must complete the initial hardware configuration. For more information, see Initial Configuration.

Configuring the HSM of your NetScaler FIPS appliance erases all existing data on the HSM. To configure the HSM, you must be logged on to the appliance as the superuser (nsroot account). The HSM is preconfigured with default values for the Security Officer (SO) password and User password, which you use to configure the HSM or reset a locked HSM. The maximum length allowed for the password is 14 alphanumeric characters. Symbols are not allowed.
Important: Do not perform the set ssl fips command without first resetting the FIPS card and restarting the MPX FIPS appliance.

Although the FIPS appliance can be used with the default password values, you should modify them before using it. The HSM can be configured only when you log on to the appliance as the superuser and specify the SO and User passwords.

Important: Due to security constraints, the appliance does not provide a means for retrieving the SO password. Store a copy of the password safely. Should you need to reinitialize the HSM, you will need to specify this password as the old SO password.

Before initializing the HSM, you can upgrade to the latest build of the software. To upgrade to the latest build, see Upgrading or Downgrading the System Software.

After upgrading, verify that the /nsconfig/fips directory has been successfully created on the appliance.

To configure the HSM on an MPX 9700/10500/12500/15500 FIPS appliances by using the command line interface

After logging on to the appliance as the superuser and completing the initial configuration, at the command prompt, type the following commands to configure the HSM and verify the configuration:

1. show ssl fips
2. reset ssl fips
3. reboot
4. set ssl fips -initHSM `Level-2` <newSOpassword> <oldSOpassword> <userPassword> [-hsmLabel <string>]
5. save ns config
6. reboot
7. show ssl fips

Example

```
show fips
FIPS Card is not configured
Done
reset fips
reboot
Are you sure you want to restart NetScaler (Y/N)? [N]:y
set ssl fips -initHSM Level-2 sopin12345 so12345 user123 -hsmLabel cavium
This command will erase all data on the FIPS card. You must save the configuration
(saveconfig) after executing this command.

Do you want to continue?(Y/N)y
Done
save ns config
reboot
Are you sure you want to restart NetScaler (Y/N)? [N]:y
show fips
        FIPS HSM Info:
HSM Label               : NetScaler FIPS
Initialization          : FIPS-140-2 Level-2
HSM Serial Number       : 2.1G1008-IC000021
HSM State               : 2
HSM Model               : NITROX XL CN1620-NFBE
Firmware Version        : 1.1
Firmware Release Date   : Jun04,2010

Max FIPS Key Memory     : 3996
Free FIPS Key Memory    : 3994
Total SRAM Memory       : 467348
Free SRAM Memory        : 62564
Total Crypto Cores       : 3
```

```
Enabled Crypto Cores    : 1
Done
```

Note: If you upgrade the firmware to version 2.2, the firmware release date is replaced with the firmware build.

```
> show fips
FIPS HSM Info:
HSM Label              : NetScaler FIPS
Initialization         : FIPS-140-2 Level-2
HSM Serial Number      : 3.0G1235-ICM000264
HSM State              : 2
HSM Model              : NITROX XL CN1620-NFBE
Hardware Version       : 2.0-G
Firmware Version       : 2.2
Firmware Build         : NFBE-FW-2.2-130009

Max FIPS Key Memory : 3996
Free FIPS Key Memory : 3958
Total SRAM Memory    : 467348
Free SRAM Memory     : 50524
Total Crypto Cores      : 3
Enabled Crypto Cores  : 3
Done
```

To configure the HSM on an MPX 9700/10500/12500/15500 FIPS appliances by using the configuration utility

1. Navigate to Traffic Management > SSL > FIPS.
2. In the details pane, on the FIPS Infotab, click Reset FIPS.
3. In the navigation pane, click System.
4. In the details pane, click Reboot.
5. In the details pane, on the FIPS Info tab, click Initialize HSM.
6. In the Initialize HSM dialog box, specify values for the following parameters:
   - Security Officer (SO) Password*â€”new SO password
   - Old SO Password*â€”old SO password
   - User Password*â€”user password
   - Levelâ€”initHSM (Currently set to Level2 and cannot be changed)
   - HSM Labelâ€”hsmLabel

   *A required parameter
7. Click OK.
8. In the details pane, click Save.
9. In the navigation pane, click System.
10. In the details pane, click Reboot.
11. Under FIPS HSM Info, verify that the information displayed for the FIPS HSM that you just configured is correct.

Creating and Transferring FIPS Keys

After configuring the HSM of your FIPS appliance, you are ready to create a FIPS key. The FIPS key is created in the applianceâ€™s HSM. You can then export the FIPS key to the applianceâ€™s CompactFlash card as a secured backup. Exporting the key also enables you to transfer it by copying it to the /flash of another appliance and then importing it into the HSM of that appliance. You must enable SIM between two standalone nodes before you export and transfer the keys. In an HA setup, if one of the nodes is replaced with a new appliance, you must enable SIM between this new appliance and the existing appliance of the HA setup before you export or import FIPS keys.

Instead of creating a FIPS key, you can import an existing FIPS key or import an external key as a FIPS key. If you are adding a certificate-key pair of 2048 bits on the MPX 9700/10500/12500/15500 FIPS appliances, make sure that you have the correct certificate and key pair.

Note: If you are planning an HA setup, make sure that the FIPS appliances are configured in an HA setup before creating a FIPS key.

Creating a FIPS Key

Before creating a FIPS key, make sure that the HSM is configured.

To create a FIPS key by using the configuration utility

1. Navigate to Traffic Management > SSL > FIPS.
2. In the details pane, on the FIPS Keys tab, click Add.
3. In the Create FIPS Key dialog box, specify values for the following parameters:
    - FIPS Key Name*â€”fipsKeyName
    - Modulus*â€”modulus
    - Exponent*â€”exponent

    *A required parameter
4. Click Create, and then click Close.
5. On the FIPS Keys tab, verify that the settings displayed for the FIPS key that you just created are correct.

To create a FIPS key by using the command line interface

At the command prompt, type the following commands to create a FIPS key and verify the settings:

- create ssl fipsKey <fipsKeyName> -modulus <positive_integer> [-exponent (3 | F4)]
- show ssl fipsKey [<fipsKeyName>]

Example

```
create fipskey Key-FIPS-1 -modulus 2048 -exponent 3
show ssl fipsKey Key-FIPS-1
FIPS Key Name: Key-FIPS-1 Modulus: 2048   Public Exponent: 3 (Hex: 0x3)
```

Exporting a FIPS Key

Citrix recommends that you create a backup of any key created in the FIPS HSM. If a key in the HSM is deleted, there is no way to create the same key again, and all the certificates associated with it are rendered useless.

In addition to exporting a key as a backup, you might need to export a key for transfer to another appliance.

The following procedure provides instructions on exporting a FIPS key to the /nsconfig/ssl folder on the appliance's CompactFlash and securing the exported key by using a strong asymmetric key encryption method.

To export a FIPS key by using the command line interface

At the command prompt, type:
export ssl fipsKey <fipsKeyName> -key <string>

Example

```
export fipskey Key-FIPS-1 -key Key-FIPS-1.key
```

To export a FIPS key by using the configuration utility

1. Navigate to Traffic Management > SSL > FIPS
2. In the details pane, on the FIPS Keys tab, click Export.
3. In the Export FIPS key to a file dialog box, specify values for the following parameters:
   - FIPS Key Name*â€”fipsKeyName
   - File Name*â€”key (To put the file in a location other than the default, you can either specify the complete path or click the Browse button and navigate to a location.)

   *A required parameter
4. Click Export, and then click Close.

Importing an Existing FIPS Key

To use an existing FIPS key with your FIPS appliance, you need to transfer the FIPS key from the hard disk of the appliance into its HSM.

Note: To avoid errors when importing a FIPS key, make sure that the name of the key imported is the same as the original key name when it was created.

To import a FIPS key on the MPX 9700/10500/12500/15500 FIPS appliances by using the command line interface

At the command prompt, type the following commands to import a FIPS key and verify the settings:

- import ssl fipsKey <fipsKeyName> -key <string> -inform `SIM` -exponent (F4 | 3)
- show ssl fipskey <fipsKeyName>

Example

```
import fipskey Key-FIPS-2 -key Key-FIPS-2.key -inform SIM -exponent F4
show ssl fipskey key-FIPS-2
FIPS Key Name: Key-FIPS-2 Modulus: 2048   Public Exponent: F4 (Hex value 0x10001)
```

To import a FIPS key by using the configuration utility

1. Navigate to Traffic Management > SSL > FIPS
2. In the details pane, on the FIPS Keys tab, click Import.
3. In the Import as a FIPS Key dialog box, select FIPS key file and set values for the following parameters:
   - FIPS Key Name*
   - Key File Name*â€”To put the file in a location other than the default, you can either specify the complete path or click Browse and navigate to a location.
   - Exponent*

   *A required parameter
4. Click Import, and then click Close.
5. On the FIPS Keys tab, verify that the settings displayed for the FIPS key that you just imported are correct.

Importing External Keys

In addition to transferring FIPS keys that are created within the NetScaler applianceâ€™s HSM, you can transfer external private keys (such as those created on a standard NetScaler, Apache, or IIS) to a FIPS NetScaler appliance. External keys are created outside the HSM, by using a tool such as OpenSSL. Before importing an external key into the HSM, copy it to the appliance's flash drive under /nsconfig/ssl.

Importing an external key as a FIPS key on the MPX 9700/10500/12500/15500 FIPS appliances by using the command line interface

On the MPX 9700/10500/12500/15500 FIPS appliances, the -exponent parameter in the import ssl fipskey command is not required while importing an external key. The correct public exponent is detected automatically when the key is imported, and the value of the -exponent parameter is ignored.

The NetScaler FIPS appliance does not support external keys with a public exponent other than 3 or F4.

You do not need a wrap key on the MPX 9700/10500/12500/15500 FIPS appliances.

You cannot import an external, encrypted FIPS key directly to an MPX 9700/10500/12500/15500 FIPS appliance. To import the key you need to first decrypt the key, and then import it. To decrypt the key, at the shell prompt, type:

```
openssl rsa -in <EncryptedKey.key> > <DecryptedKey.out>
```

To import an external key as a FIPS key to an MPX 9700/10500/12500/15500 FIPS appliance by using the command line interface

1. Copy the external key to the appliance's flash drive.
2. If the key is in .pfx format, you must first convert it to PEM format. At the command prompt, type:
   - convert ssl pkcs12 <output file> -import -pkcs12File <input .pfx file name> -password <password>
3. At the command prompt, type the following commands to import the external key as a FIPS key and verify the settings:
   - import ssl fipsKey <fipsKeyName> -key <string> -informPEM
   - show ssl fipskey<fipsKeyName>

Example

```
convert ssl pkcs12 iis.pem -password 123456 -import -pkcs12File iis.pfx
import fipskey Key-FIPS-2 -key iis.pem -inform PEM
show ssl fipskey key-FIPS-2
FIPS Key Name: Key-FIPS-2 Modulus: 0   Public Exponent: F4 (Hex value 0x10001)
```

Note: The modulus is incorrectly displayed as zero in the above example. The discrepancy does not affect SSL functionality.

To import an external key as a FIPS key to an MPX 9700/10500/12500/15500 FIPS appliance by using the configuration utility

1. If the key is in .pfx format, you must first convert it to PEM format.
   a. Navigate to Traffic Management > SSL.
   b. In the details pane, under Tools, click Import PKCS#12.
   c. In the Import PKCS12 File dialog box, set the following parameters:
      - Output File Name*
      - PKCS12 File Name*â€”Specify the .pfx file name.
      - Import Password*
      - Encoding Format

   *A required parameter
2. Navigate to Traffic Management > SSL > FIPS
3. In the details pane, on the FIPS Keys tab, click Import.
4. In the Import as a FIPS Key dialog box, select PEM file, and set values for the following parameters:
   - FIPS Key Name*
   - Key File Name*â€”To put the file in a location other than the default, you can either specify the complete path or click Browse and navigate to a location.

   *A required parameter
5. Click Import, and then click Close.
6. On the FIPS Keys tab, verify that the settings displayed for the FIPS key that you just imported are correct.

Configuring FIPS Appliances in a High Availability Setup

You can configure two appliances in a high availability (HA) pair as FIPS appliances. For information about configuring an HA setup, see .

Note: Citrix recommends that you use the configuration utility (GUI) for this procedure. If you use the command line (CLI), make sure that you carefully follow the steps as listed in the procedure. Changing the order of steps or specifying an incorrec input file might cause inconsistency that requires that the appliance be restarted. In addition, if you use the CLI, the create ssl fipskey command is not propagated to the secondary node. When you execute the command with the same input values for modulus size and exponent on two different FIPS appliances, the keys generated are not identical. You have to create the FIPS key on one of the nodes and then transfer it to the other node. But if you use the configuration utility to configure FIPS appliances in an HA setup, the FIPS key that you create is automatically transferred to the secondary node. The process of managing and transferring the FIPS keys is known as secure information management (SIM).
Important: On the MPX 9700/10500/12500/15500 FIPS appliances, the HA setup should be completed within six minutes. If the process takes longer than six minutes, the internal timer of the FIPS card expires and the following error message appears:

ERROR: Operation timed out or repeated, please wait for 10 mins and redo the SIM/HA configuration steps.

If this message appears, restart the appliance or wait for 10 minutes, and then repeat the HA setup procedure.

In the following procedure, appliance A is the primary node and appliance B is the secondary node.

To configure FIPS appliances in a high availability setup by using the command line interface

1. **On appliance A**, open an SSH connection to the appliance by using an SSH client, such as PuTTY.
2. Log on to the appliance, using the administrator credentials.
3. Initialize appliance A as the source appliance. At the command prompt, type:

   ```
   init ssl fipsSIMsource <certFile>
   ```
4. Copy this <certFile> file to appliance B, in the /nconfig/ssl folder.
5. **On appliance B**, open an SSH connection to the appliance by using an SSH client, such as PuTTY.
6. Log on to the appliance, using the administrator credentials.
7. Initialize appliance B as the target appliance. At the command prompt, type:

   ```
   init ssl fipsSIMtarget <certFile> <keyVector> <targetSecret>
   ```
8. Copy this <targetSecret> file to appliance A.
9. **On appliance A**, enable appliance A as the source appliance. At the command prompt, type:

   ```
   enable ssl fipsSIMSource <targetSecret> <sourceSecret>
   ```
10. Copy this <sourceSecret> file to appliance B.
11. **On appliance B**, enable appliance B as the target appliance. At the command prompt, type:

    ```
    enable ssl fipsSIMtarget <keyVector> <sourceSecret>
    ```
12. **On appliance A**, create a FIPS key, as described in Creating a FIPS Key.
13. Export the FIPS key to the applianceâ€™s hard disk, as described in Exporting a FIPS Key.
14. Copy the FIPS key to the hard disk of the secondary appliance by using a secure file transfer utility, such as SCP.
15. **On appliance B**, import the FIPS key from the hard disk into the HSM of the appliance, as described in Importing an Existing FIPS Key.

To configure FIPS appliances in a high availability setup by using the configuration utility

1. On the appliance to be configured as the source appliance, navigate to Traffic Management > SSL > FIPS.
2. In the details pane, on the FIPS Info tab, click Enable SIM.
3. In the Enable HA Pair for SIM dialog box, in the Certificate File Name text box, type the file name, with the path to the location at which the FIPS certificate should be stored on the source appliance.
4. In the Key Vector File Name text box, type the file name, with the path to the location at which the FIPS key vector should be stored on the source appliance.
5. In the Target Secret File Name text box, type the location for storing the secret data on the target appliance.
6. In the Source Secret File Name text box, type the location for storing the secret data on the source appliance.
7. Click OK. The FIPS appliances are now configured in HA mode.
8. Create a FIPS key, as described in Creating a FIPS Key. The FIPS key is automatically transferred from the primary to the secondary.

Example

In the following example, source.cert is the certificate on the source appliance, stored in the default directory, /nsconfig/ssl. This certificate must be transferred to the same location (/nsconfig/ssl) on the target appliance. The file target.secret is created on the target appliance and copied to the source appliance. The file source.secret is created on the source appliance and copied to the target appliance.

On the source appliance

```
init fipsSIMsource /nsconfig/ssl/source.cert
```

On the target appliance

```
init fipsSIMtarget /nsconfig/ssl/source.cert /nsconfig/ssl/target.key /nsconfig/ssl/target.
```

On the source appliance

```
enable fipsSIMsource /nsconfig/ssl/target.secret /nsconfig/ssl/source.secret
```

On the target appliance

```
enable fipsSIMtarget /nsconfig/ssl/target.key /nsconfig/ssl/source.secret
```

On the source appliance

```
create ssl fipskey fips1 -modulus 2048 -exponent f4
export fipskey fips1 -key /nsconfig/ssl/fips1.key
```

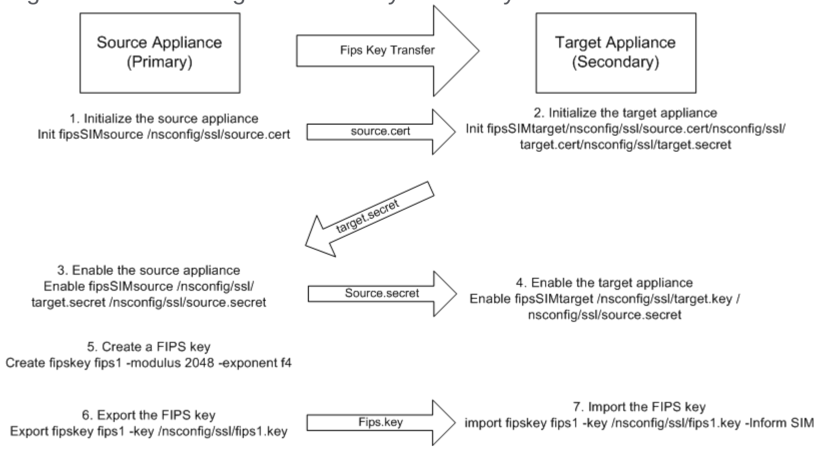
Copy this key into the hard disk of the target appliance.

On the target appliance

```
import fipskey fips1 -key /nsconfig/ssl/fips1.key
```

The following diagram summarizes the transfer process.

Figure 1. Transferring the FIPS Key-Summary


Source Appliance (Primary)
Fips Key Transfer
Target Appliance (Secondary)
1. Initialize the source appliance
Init fipsSIMsource /nsconfig/ssl/source.cert
source.cert
2. Initialize the target appliance
Init fipsSIMtarget/nsconfig/ssl/source.cert/nsconfig/ssl/target.cert/nsconfig/ssl/target.secret
target.secret
3. Enable the source appliance
Enable fipsSIMsource /nsconfig/ssl/target.secret /nsconfig/ssl/source.secret
Source.secret
4. Enable the target appliance
Enable fipsSIMtarget /nsconfig/ssl/target.key /nsconfig/ssl/source.secret
5. Create a FIPS key
Create fipskey fips1 -modulus 2048 -exponent f4
6. Export the FIPS key
Export fipskey fips1 -key /nsconfig/ssl/fips1.key
Fips.key
7. Import the FIPS key
import fipskey fips1 -key /nsconfig/ssl/fips1.key -Inform SIM

Updating the Firmware to Version 2.2 on a FIPS Card

FIPS firmware version 2.2 supports TLS protocol versions 1.1 and 1.2. From the command line, you can update the firmware version of the FIPS card of a NetScaler MPX 9700/10500/12500/15500 FIPS appliance from version 1.1 to version 2.2.

For successful SIM key propagation from primary to secondary in a high availability (HA) pair, the Cavium firmware version on each appliance should be identical. Perform the firmware update on the secondary appliance first. If executed on the primary appliance first, the long-running update process causes a failover.

Limitations

- Secure renegotiation is supported only on SSL virtual servers and front-end SSL services.
- Creating a certificate signing request by using a key that was created on firmware version 1.1 and updated to firmware version 2.2 fails.
- You cannot create a 1024-bit RSA key on firmware version 2.2. However, if you have imported or created a 1024-bit FIPS key on firmware version 1.1 and you then update to firmware version 2.2, you can use that FIPS key on firmware version 2.2.
- 1024-bit RSA keys are not supported.
- Secure renegotiation using SSLv3 protocol is not supported.
- After you upgrade the firmware, TLSv1.1 and TLSv1.2 are disabled by default on the existing virtual server, internal, front end, and backend services. To use TLS 1.1/1.2, you must explicitly enable these protocols, on the SSL entities, after the upgrade.
- FIPS keys that are created in firmware version 2.2 are not available if you downgrade the firmware to version 1.1.

Prerequisites

Download the following files from the download page on www.citrix.com. The files must be stored in the /var/nsinstall directory on the appliance.

- FW 2.2 File: FW-2.2-130013
- FW 2.2 Signature File: FW-2.2-130013.sign

To update the FIPS firmware to version 2.2 on a standalone appliance

1. Log on to the appliance by using the administrator credentials.
2. At the prompt, type the following command to confirm that the FIPS card is initialized.

   show fips

```
FIPS HSM Info:
                HSM Label                : NetScaler FIPS
                Initialization                : FIPS-140-2 Level-2
                HSM Serial Number        : 3.0G1235-ICM000264
                HSM State                : 2
                HSM Model                : NITROX XL CN1620-NFBE

                Hardware Version        : 2.0-G
                Firmware Version        : 1.1
                Firmware Release Date        : Jun04,2010

                Max FIPS Key Memory        : 3996
                Free FIPS Key Memory        : 3992
                Total SRAM Memory        : 467348
                Free SRAM Memory        : 62512
                Total Crypto Cores        : 3
                Enabled Crypto Cores        : 1
         Done
```

3. Save the configuration. At the prompt, type:

   save config

4. Perform the update. At the prompt, type:

update ssl fips -fipsFW <path to the extracted contents>/CN16XX-NFBE-FW-2.2-1300013

and press Y when the following prompt appears:

```
This command will update compatible version of the FIPS firmware.  You must save the cu
Done
```

Note: You only need to specify the firmware file, because the firmware signature file is placed in the same location.

The update takes up to ten seconds. The update command is blocking, which means that no other actions are executed until the command finishes. The command prompt reappears when execution of the command is completed.

5. Restart the appliance. At the prompt, type:

   reboot

   ```
   Are you sure you want to restart NetScaler (Y/N)? [N]:Y
   ```
6. Verify that the update is successful. At the prompt, type:

   show fips

   The firmware version displayed in the output should be 2.2. For example:

   ```
   > sh fips
           FIPS HSM Info:
                   HSM Label                : NetScaler FIPS
                   Initialization                : FIPS-140-2 Level-2
                   HSM Serial Number        : 2.1G1207-IC002429
                   HSM State                : 2
                   HSM Model                : NITROX XL CN1620-NFBE

                   Hardware Version        : 2.0-G
                   Firmware Version        : 2.2
                   Firmware Build              : NFBE-FW-2.2-130013
                   Max FIPS Key Memory        : 3996
                   Free FIPS Key Memory         : 3982
                   Total SRAM Memory        : 467348
                   Free SRAM Memory        : 50472
                   Total Crypto Cores        : 3
                   Enabled Crypto Cores         : 1
    Done
   ```

To update the FIPS firmware to version 2.2 on appliances in a high availability pair

1. Log on to the secondary node and perform the update as described in To update the FIPS firmware to version 2.2 on a standalone NetScaler.

   Force the secondary node to become primary. At the prompt, type:

   force failover

   and press **Y** at the confirmation prompt.
2. Log on to the new secondary node (old primary) and perform the update as described in To update the FIPS firmware to version 2.2 on a standalone NetScaler.
3. Force the new secondary node to become primary again. At the prompt, type:

   force failover

   and press **Y** at the confirmation prompt.

To update the FIPS firmware to version 1.1 on a standalone appliance

1. Download the nfb_firmware-r1235_100604 and nfb_firmware-r1235_100604.sign files, to the same directory on the appliance, from the download page on www.citrix.com.

2. Log on to the appliance by using the administrator credentials.

3. At the prompt, type:

```
update ssl fips -fipsFW /<full path to the file>/nfb_firmware-r1235_100604
```

Resetting a Locked HSM

The HSM becomes locked (no longer operational) if you change the SO password, restart the appliance without saving the configuration, and make three unsuccessful attempts to change the password. This is a security measure for preventing unauthorized access attempts and changes to the HSM settings.

Important: To avoid this situation, save the configuration after initializing the HSM.

If the HSM is locked, you must reset the HSM and restart the appliance to restore the default passwords. You can then use the default passwords to access the HSM and configure it with new passwords. When finished, you must save the configuration and restart the appliance.

Caution: Do not reset the HSM unless it has become locked.

To reset a locked HSM by using the command line interface

At the command prompt, type the following commands to reset and re-initialize a locked HSM:

- reset ssl fips
- reboot -warm
- set ssl fips -initHSM `Level-2` <new SO password> <old SO password> <user password> [-hsmLabel <string>]
- save ns config
- reboot -warm

Example

```
reset fips
reboot -warm
set fips -initHSM Level-2 newsopin123 sopin123 userpin123 -hsmLabel NSFIPS
saveconfig
reboot -warm
```

Note: The SO and User passwords are the default passwords.

To reset a locked HSM by using the configuration utility

1. Navigate to Traffic Management > SSL > FIPS
2. In the details pane, on the FIPS Info tab, click Reset FIPS.
3. Configure the HSM, as described in Configuring the HSM.
4. In the details pane, click Save.

FIPS Approved Algorithms and Ciphers

Table 1: Ciphers supported on a NetScaler FIPS Appliance

Cipher Suite	Protocol	Key Exchange Algorithm	Authentication Algorithm	Encryption Algorithm (Key Size)	Message Authentic Code (MA Algorithm
TLS1-AES-256-CBC-SHA	SSLv3	RSA	RSA	AES(256)	SHA1
TLS1-AES-128-CBC-SHA	SSLv3	RSA	RSA	AES(128)	SHA1
TLS1-ECDHE-RSA-AES256-SHA	SSLv3	ECC-DHE	RSA	AES(256)	SHA1
TLS1-ECDHE-RSA-AES128-SHA	SSLv3	ECC-DHE	RSA	AES(128)	SHA1
TLS1.2-ECDHE-RSA-AES-256-SHA384 (frontend only)	TLSv1.2	ECC-DHE	RSA	AES(256)	SHA-384
TLS1.2-ECDHE-RSA-AES-128-SHA256	TLSv1.2	ECC-DHE	RSA	AES(128)	SHA-256
TLS1-ECDHE-RSA-DES-CBC3-SHA	SSLv3	ECC-DHE	RSA	3DES (168)	SHA1
SSL3-DES-CBC3-SHA	SSLv3	RSA	RSA	3DES (168)	SHA1

For information about the Cipher/Protocol support matrix, see http://docs.citrix.com/en-us/netscaler/11-1/ssl/cipher_protocl_support_matrix.html.

Support for Thales nShieldÂ® HSM

Note: This feature is available from release 11, build 62.10.

A non-FIPS NetScaler appliance stores the serverâ€™s private key on the hard disk. On a FIPS appliance, the key is stored in a cryptographic module known as hardware security module (HSM). Storing a key in the HSM protects it from physical and software attacks. In addition, the keys are encrypted by using special FIPS approved ciphers.

Only the NetScaler MPX 9700/10500/12500/15500 FIPS appliances support a FIPS card. Support for FIPS is not available on other MPX appliances, or on the SDX and VPX appliances. This limitation is addressed by supporting a Thales nShieldÂ® Connect external HSM on all NetScaler MPX, SDX, and VPX appliances except the MPX 9700/10500/12500/15500 FIPS appliances.

Thales nShield Connect is an external FIPS-certified network-attached HSM. With a Thales HSM, the keys are securely stored as application key tokens on a remote file server (RFS) and can be reconstituted inside the Thales HSM only.

If you are already using a Thales HSM, you can now use a NetScaler ADC to optimize, secure, and control the delivery of all enterprise and cloud services.

Note:

- Thales HSMs comply with FIPS 140-2 Level 3 specifications, while the MPX FIPS appliances comply with level 2 specifications.
- You cannot decrypt the trace while using the Thales HSM, because the response from the HSM to the NetScaler appliance is encrypted and only the Hardserver can read it.

This section includes the following details:

Architecture Overview

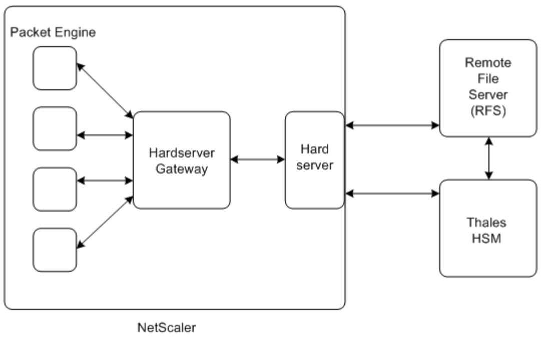
The three entities that are part of a NetScaler-Thales deployment are a Thales nShield Connect module, a remote file server (RFS), and a NetScaler ADC.

The Thales nShield Connect is a network-attached hardware security module. The RFS is used to configure the HSM and to store the encrypted key files.

Hardserver, a proprietary daemon provided by Thales, is used for communication between the client (ADC), the Thales HSM, and the RFS. It uses the IMPATH secure communication protocol. A gateway daemon, called the Hardserver Gateway, is used to communicate between the NetScaler packet engine and the Hardserver.

Note: The terms Thales nShield Connect, Thales HSM, and HSM are used interchangeably in this documentation.

The following figure illustrates the interaction between the different components.

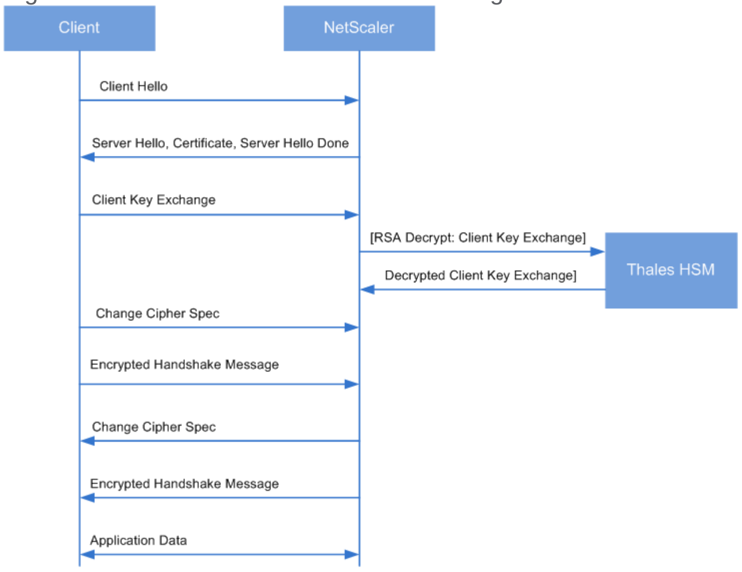


In a typical deployment, the RFS is used to securely store keys generated by the HSM. After the keys are generated, you can securely transfer them to the ADC and then use the NetScaler configuration utility or command line to load the keys to the HSM. A virtual server on the ADC uses Thales to decrypt the client key exchange to complete the SSL handshake. Thereafter, all the SSL operations are performed on the ADC.

Note: The terms keys and application key tokens are used interchangeably in this documentation.

The following figure illustrates the packet flow in the SSL handshake with the Thales HSM.

Figure 1. SSL Handshake Packets Flow Diagram with NetScaler Using Thales HSM


Client
NetScaler
Client Hello
Server Hello, Certificate, Server Hello Done
Client Key Exchange
[RSA Decrypt: Client Key Exchange]
Decrypted Client Key Exchange]
Thales HSM
Change Cipher Spec
Encrypted Handshake Message
Change Cipher Spec
Encrypted Handshake Message
Application Data

Note: The communication between the ADC and the HSM uses a Thales proprietary communication protocol, called IMPATH

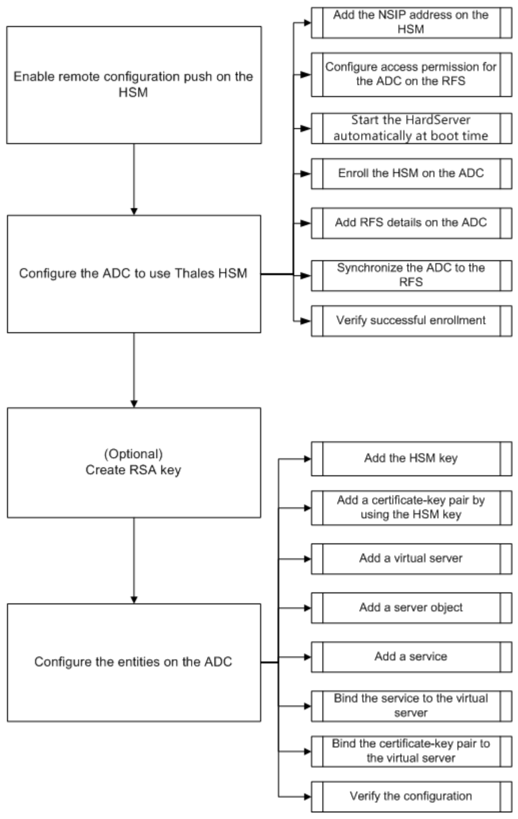
Prerequisites

Before you can use a Thales nShield Connect with a NetScaler ADC, make sure that the following prerequisites are met:

- A Thales nShield Connect device is installed in the network, ready to use, and accessible to the NetScaler ADC. That is, the NetScaler IP (NSIP) address is added as an authorized client on the HSM.
- A usable Security World exists. Security World is a unique key management architecture used by the Thales nShield line of HSMs. It protects and manages keys as application key tokens, enabling unlimited key capacity, and automatic key backup and recovery. For more information about creating a Security World, see the nShield Connect Quick Start Guide from Thales. You can also find the guide in the CD provided with the Thales HSM module at CipherTools-linux-dev-xx.xx.xx/document/nShield_Connect_Quick_Start_Guide.pdf.
  Note: Softcard or token/OCS protected keys are currently not supported on the NetScaler ADC.
- Licenses are available to support the number of clients that will be connected to the Thales HSM. The ADC and RFS are clients of the HSM.
- A remote file server (RFS) is installed in the network and is accessible to the NetScaler ADC.
- The Thales nShield Connect device, the RFS, and the NetScaler ADC can initiate connections with each other through port 9004.
- You are using NetScaler release 10.5 build 52.1115.e or later.
- The NetScaler appliance does not contain a FIPS Cavium card.
  Important: Thales HSM is not supported on the MPX 9700/10500/12500/15500 FIPS appliances.

Configuring the ADC-Thales Integration

The following flowchart depicts the tasks that you need to perform to use Thales HSM with a NetScaler ADC:


Enable remote configuration push on the HSM
Add the NSIP address on the HSM
Configure access permission for the ADC on the RFS
Start the HardServer automatically at boot time
Enroll the HSM on the ADC
Add RFS details on the ADC
Configure the ADC to use Thales HSM
Synchronize the ADC to the RFS
Verify successful enrollment
(Optional) Create RSA key
Add the HSM key
Add a certificate-key pair by using the HSM key
Add a virtual server
Add a server object
Configure the entities on the ADC
Add a service
Bind the service to the virtual server
Bind the certificate-key pair to the virtual server
Verify the configuration

As shown in the above flowchart, you perform the following tasks:

1. Enable remote configuration push on the HSM.
2. Configure the ADC to use the Thales HSM.
   - Add the NSIP address on the HSM.
   - Configure access permission for the ADC on the RFS.
   - Configure automatic start of the Hardserver at boot time.
   - Enroll the HSM on the ADC.
   - Add RFS details on the ADC.
   - Synchronize the ADC to the RFS.
   - Verify that Thales HSM is successfully enrolled on the ADC.
3. (Optional) Create an HSM RSA key.
4. Configure the entities on the NetScaler ADC.
   - Add the HSM key.
   - Add a certificate-key pair by using the HSM key.
   - Add a virtual server.
   - Add a server object.
   - Add a service.
   - Bind the service to the virtual server.
   - Bind the certificate-key pair to the virtual server.
   - Verify the configuration.

1. Configure the Thales HSM

You must specify the IP address of the RFS on the Thales HSM so that it accepts the configuration that the RFS pushes to it. Use the nShield Connect front panel on the Thales HSM to perform the following procedure.

To specify the IP address of a remote computer on the Thales HSM

1. Navigate to System Configuration > Config file options > Allow auto push.
2. Select ON, and specify the IP address of the computer (RFS) from which to accept the configuration.

Enabling Remote Configuration Push on the HSM

You must specify the IP address of the RFS on the Thales HSM so that it accepts the configuration that the RFS pushes to it. Use the nShield Connect front panel on the Thales HSM to perform the following procedure.

To specify the IP address of a remote computer on the Thales HSM

1. Navigate to System Configuration > Config file options > Allow auto push.
2. Select ON, and specify the IP address of the computer (RFS) from which to accept the configuration.

Configure the ADC to use the Thales HSM

Sample values used in this documentation:

NSIP address=10.217.2.43

Thales HSM IP address=10.217.2.112

RFS IP address=10.217.2.6

Add the NetScaler IP (NSIP) Address on the HSM

Typically you use the nShield Connect front panel to add clients to the HSM. For more information, see the nShield Connect Quick Start Guide.

Alternately, use the RFS to add the ADC as a client to the HSM. To do this, you must add the NSIP address in the HSM configuration on the RFS, and then push the configuration to the HSM. Before you can do this, you must know the electronic serial number (ESN) of the HSM.

To get the ESN of your HSM, run the following command on the RFS:

```
root@ns# /opt/nfast/bin/anonkneti <Thales HSM IP address>
```

Example

```
root@ns# /opt/nfast/bin/anonkneti 10.217.2.112
BD17-C807-58D9 5e30a698f7bab3b2068ca90a9488dc4e6c78d822
```

The ESN number is BD17-C807-58D9.

After you have the ESN number, use an editor, such as vi, to edit the HSM configuration file on the RFS.

```
vi /opt/nfast/kmdata/hsm-BD17-C807-58D9/config/config
```

In the `hs_clients` section, add the following entries:

```
# Amount of data in bytes to encrypt with a session key before session key# renegotiation, or 0
#  datalimit=INT
addr=10.217.2.43
clientperm=unpriv
keyhash=0000000000000000000000000000000000000000
esn=
timelimit=86400
datalimit=8388608
-----
```

Note: Include one or more hyphens as delimiters to add multiple entries in the same section.

To push the configuration to the HSM, run the following command on the RFS:

```
/opt/nfast/bin/cfg-pushnethsm --address=<Thales HSM IP address> --force
/opt/nfast/kmdata/hsm-BD17-C807-58D9/config/config
```

Example

```
/opt/nfast/bin/cfg-pushnethsm --address=10.217.2.112 --force
                                /opt/nfast/kmdata/hsm-BD17-C807-58D9/config/config
```

Configure Access Permission for the ADC on the RFS

To configure access permission for the ADC on the RFS, run the following command on the RFS:

```
/opt/nfast/bin/rfs-setup --force -g --write-noauth <NetScaler IP address>
```

Example

```
[root@localhost bin]# /opt/nfast/bin/rfs-setup --force -g --write-noauth 10.217.2.43
Adding read-only remote_file_system entries
Ensuring the directory /opt/nfast/kmdata/local exists
Adding new writable remote_file_system entries
Ensuring the directory /opt/nfast/kmdata/local/sync-store exists
Saving the new config file and configuring the hardserver
Done
```

Verify that the ADC can reach both the RFS and Thales HSM by using port 9004.

Configure Automatic Start of the Hardserver at Boot Time

Create a file and then restart the appliance. Now, whenever you restart the appliance, and if this file is found, the Hardserver is automatically started.

At the shell prompt, type:
touch /var/opt/nfast/bin/thales_hsm_is_enrolled

At the command prompt, type:
reboot

Enroll the HSM on the ADC

Change directory to /var/opt/nfast/bin.

To add HSM details into the ADC configuration, run the following command on the ADC:

```
nethsmenroll --force <Thales_nShield_Connect_ip_address> $(anonkneti
<Thales_nShield_Connect_ip_address>)
```

Example

```
root@ns# ./nethsmenroll --force 10.217.2.112 $(anonkneti 10.217.2.112)
OK configuring hardserver's nethsm imports
```

This step adds the following entries after the line `# ntoken_esn=ESN` in the `nethsm_imports` section of the /var/opt/nfast/kmdata/config/config file.

```
â€¦
local_module=0
remote_ip=10.217.2.112
remote_port=9004
remote_esn=BD17-C807-58D9
keyhash=5e30a698f7bab3b2068ca90a9488dc4e6c78d822
timelimit=86400
datalimit=8388608
privileged=0
privileged_use_high_port=0
ntoken_esn=
```

Change directory to /var/opt/nfast/bin and run the following command on the ADC:
touch "thales_hsm_is_enrolled"
Note: To remove an HSM that is enrolled on the ADC, type: `./nethsmenroll â€“-remove <NETHSM-IP>`

Add RFS details on the ADC

To add RFS details, change directory to /var/opt/nfast/bin/ and then run the following command:

```
./rfs-sync --no-authenticate --setup <rfs_ip_address>
```

Example

```
./rfs-sync --no-authenticate --setup 10.217.2.6
No current RFS synchronization configuration.
Configuration successfully written; new config details:
Using RFS at 10.217.2.6:9004: not authenticating.
```

This step adds the following entries after the `# local_esn=ESN` line in the `rfs_sync_client` section of the /var/opt/nfast/kmdata/config/config file.

```
â€¦â€¦
remote_ip=10.217.2.6
remote_port=9004
use_kneti=no
local_esn=
```

Note: To remove an RFS that is enrolled on the ADC, type: `./rfs_sync â€“remove`
Example

```
./rfs-sync --no-authenticate --setup 10.217.2.6
No current RFS synchronization configuration.
Configuration successfully written; new config details:
Using RFS at 10.217.2.6:9004: not authenticating.
```

This step adds the following entries after the `# local_esn=ESN` line in the `rfs_sync_client` section of the /var/opt/nfast/kmdata/config/config file.

```
â€¦â€¦
remote_ip=10.217.2.6
remote_port=9004
use_kneti=no
local_esn=
```

Synchronize the ADC to the RFS

To synchronize all the files, change directory to /var/opt/nfast/bin and then run the following command on the ADC:

```
./rfs-sync â€“-update
```

This command fetches all the World files, module files, and key files from the /opt/nfast/kmdata/local directory on the RFS and puts them into the /var/opt/nfast/kmdata/local directory on the ADC. Citrix recommends that you manually copy the World files, the module_XXXX_XXXX_XXXX files, where XXXX_XXXX_XXXX is the ESN of the enrolled HSM, and only the required RSA key and certificate files.

Verify that the Thales HSM is successfully enrolled on the ADC

After you synchronize the ADC to the RFS, do the following:

- Verify that the local Hardserver is UP and running. (`nCipher server running`).
- Get the state of the configured HSMs, and verify that the values for the `n_modules` (number of modules) field and the km info fields are non-zero.
- Verify that the HSM is enrolled correctly and is usable (`state 0x2 Usable`) by the ADC.
- Load tests using sigtest run properly.

Change directory to /var/opt/nfast/bin, and at the shell prompt, run the following commands:

```
root@ns# ./chkserv root@ns# ./nfkminfo root@ns# ./sigtest
```

See Appendix for an example.

Create an HSM RSA Key

Only RSA keys are supported as HSM keys.

Note: Skip this step if keys are already present in the /opt/nfast/kmdata/local folder on the RFS.

Create an RSA key, a self-signed certificate, and a Certificate Signing Request (CSR). Send the CSR to a certificate authority to get a server certificate.

The following files are created in the example below:

- Embed RSA key: key_embed_2ed5428aaeae1e159bdbd63f25292c7113ec2c78
- Self-Signed Certificate: example_selfcert
- Certificate Signing Request: example_req

Note: The `generatekey` command is supported in strict FIPS 140-2 Level 3 Security World. An administrator card set (ACS) or an operator card set (OCS) is needed to control many operations, including the creation of keys and OCSs. When you run the `generatekey` command, you are prompted to insert an ACS or OCS card. For more information about strict FIPS 140-2 Level 3 Security World, see the nShield Connect User Guide.

The following example uses Level-2 Security World. In the example, the commands are in boldface type.

Example

```
[root@localhost bin]# ./generatekey embed
size: Key size? (bits, minimum 1024) [1024] > 2048
OPTIONAL: pubexp: Public exponent for RSA key (hex)? []
>
embedsavefile: Filename to write key to? []
> example
plainname: Key name? [] > example
x509country: Country code? [] > US
x509province: State or province? [] > CA
x509locality: City or locality? [] > Santa Clara
x509org: Organisation? [] > Citrix
x509orgunit: Organisation unit? [] > NS
x509dnscommon: Domain name? [] > www.citrix.com
x509email: Email address? [] > example@citrix.com
nvram: Blob in NVRAM (needs ACS)? (yes/no) [no] >
digest: Digest to sign cert req with? (md5, sha1, sha256, sha384, sha512)
  [default sha1] > sha512
key generation parameters:
 operation      Operation to perform               generate
 application    Application                        embed
 verify         Verify security of key             yes
 type           Key type                           RSA
 size           Key size                           2048
 pubexp         Public exponent for RSA key (hex)
 embedsavefile  Filename to write key to           example
 plainname      Key name                           example
 x509country    Country code                       US
 x509province   State or province                  CA
 x509locality   City or locality                   Santa Clara
 x509org        Organisation                       Citrix
 x509orgunit    Organisation unit                  NS
 x509dnscommon  Domain name                        www.citrix.com
 x509email      Email address                      example@citrix.com
 nvram          Blob in NVRAM (needs ACS)          no
 digest         Digest to sign cert req with       sha512
Key successfully generated.
Path to key: /opt/nfast/kmdata/local/key_embed_2ed5428aaeae1e159bdbd63f25292c7113ec2c78
You have new mail in /var/spool/mail/root
```

Result

You have created a CSR (example_req), a self-signed certificate (example_selfcert), and an application key token file in embed format (/opt/nfast/kmdata/local/key_embed_2ed5428aaeae1e159bdbd63f25292c7113ec2c78)

Because the ADC supports keys in simple format only, you must convert the embed key to a simple key.

To convert the embed key to a simple key, run the following command on the RFS:

```
[root@localhost bin]# ./generatekey -r simple
from-application: Source application? (embed, simple) [embed] > embed
from-ident: Source key identifier? (c6410ca00af7e394157518cb53b2db46ff18ce29,
                                    2ed5428aaeae1e159bdbd63f25292c7113ec2c78)
  [default c6410ca00af7e394157518cb53b2db46ff18ce29]
> 2ed5428aaeae1e159bdbd63f25292c7113ec2c78
ident: Key identifier? [] > examplersa2048key
plainname: Key name? [] > examplersa2048key
key generation parameters:
 operation         Operation to perform    retarget
 application       Application             simple
 verify            Verify security of key  yes
 from-application  Source application      embed
 from-ident        Source key identifier   2ed5428aaeae1e159bdbd63f25292c7113ec2c78
 ident             Key identifier          examplersa2048key
 plainname         Key name                examplersa2048key
Key successfully retargetted.
Path to key: /opt/nfast/kmdata/local/key_simple_examplersa2048key
```

Important

When prompted for the source key identifier, enter **2ed5428aaeae1e159bdbd63f25292c7113ec2c78** as the embed key.

Result

A key with the prefix key_simple (for example key_simple_examplersa2048key) is created.

Note: examplersa2048key is the key identifier (ident) and is referred to as the HSM key name on the ADC. A key identifier is unique. All the simple files have the prefix key_simple.

Configure the Entities on the ADC

Before the ADC can process traffic, you must do the following:

1. Enable features.
2. Add a subnet IP (SNIP) address.
3. Add the HSM key to the ADC.
4. Add a certificate-key pair by using the HSM key.
5. Add a virtual server.
6. Add a server object.
7. Add a service.
8. Bind the service to the virtual server.
9. Bind the certificate-key pair to the virtual server.
10. Verify the configuration.

Enable features on the ADC

Licenses must be present on the ADC before you can enable a feature.

To enable a feature by using the command line

At the command prompt, run the following commands:

- enable feature `lb`
- enable feature `ssl`

To enable a feature by using the configuration utility

Navigate to System > Settings and, in the Modes and Features group, select Configure basic features, and then select SSL Offloading.

Add a subnet IP address

For more information about subnet IP addresses, see Configuring Subnet IP Addresses.

To add a SNIP address and verify the configuration by using the command line

At the command prompt, run the following commands:

- add ns ip <IPAddress> <netmask> -type `SNIP`
- show ns ip

Example

```
> add ns ip 192.168.17.253 255.255.248.0 -type SNIP
Done
> show ns ip
        Ipaddress        Traffic Domain  Type             Mode     Arp      Icmp     Vserver
        ---------        --------------  ----             ----     ---      ----     -------
1)      192.168.17.251   0               NetScaler IP     Active   Enabled  Enabled  NA
2)      192.168.17.252   0               VIP              Active   Enabled  Enabled  Enabled
3)      192.168.17.253   0               SNIP              Active   Enabled  Enabled  NA
 Done
```

To add a SNIP address and verify the configuration by using the configuration utility

Navigate to System > Network > IPs, add an IP address, and select the IP Type as `Subnet IP`.

Copy the HSM key and certificate to the ADC

Use a secure file transfer utility to securely copy the key (key_simple_examplersa2048key) to the /var/opt/nfast/kmdata/local folder, and the certificate (example_selfcert) to the /nsconfig/ssl folder on the ADC.

Add the key on the ADC

All the keys have a key-simple prefix. When adding the key to the ADC, use the ident as the HSM key name. For example, if the key that you added is key_simple_XXXX, the HSM key name is XXXX.
Important:

- The HSM key name must be the same as the ident that you specified when you converted an embed key to a simple key format.
- The keys must be present in the /var/opt/nfast/kmdata/local/ directory on the ADC.

To add an HSM key by using the command line

At the shell prompt, run the following command:

add ssl hsmKey <hsmKeyName> -key <string>

Example

```
> add ssl hsmKey examplersa2048key â€"key key_simple_examplersa2048key
Done
```

To add an HSM key by using the configuration utility

Navigate to Traffic Management > SSL > HSM, and add an HSM key.

Add a certificate-key pair on the ADC

For information about certificate-key pairs, see Adding or Updating a Certificate-Key Pair.

To add a certificate-key pair by using the command line

At the command prompt, run the following command:

add ssl certKey <certkeyName> -cert <string> -hsmKey <string>

Example

```
> add ssl certKey key22 -cert example_selfcert -hsmKey examplersa2048key
Done
```

To add a certificate-key pair by using the configuration utility

Navigate to Traffic Management > SSL > Certificates, and add a certificate-key pair.

Add a virtual server

For information about a virtual server, see Configuring an SSL-Based Virtual Server.

To configure an SSL-based virtual server by using the command line

At the command prompt, run the following command:

add lb vserver <name> <serviceType> <IPAddress> <port>

Example

```
> add lb vserver v1 SSL 192.168.17.252 443
```

To configure an SSL-based virtual server by using the configuration utility

Navigate to Traffic Management > Load Balancing > Virtual Servers, create a virtual server, and specify the protocol as `SSL`.

Add a server object

Before you can add a server object on the ADC, make sure that you have created a backend server. The following example uses the built-in python HTTP Server module on a Linux system.

Example

```
%python â€“m SimpleHTTPServer 80
```

To add a server object by using the command line

At the command prompt, run the following command:

add server <name> <IPAddress>
Example

```
> add server s1 192.168.17.246
```

To add a server object by using the configuration utilty

Navigate to Traffic Management > Load Balancing > Servers, and add a server.

Add a service

For more information, see Configuring Services.

To configure a service by using the command line

At the command prompt, run the following command:

add service <name> <serverName> <serviceType> <port>
Example

```
> add service sr1 s1 HTTP 80
```

To configure a service by using the configuration utility

Navigate to Traffic Management > Load Balancing > Services, and create a service.

Bind the service to the virtual server

For more information, see Binding Services to an SSL-Based Virtual Server.

To bind a service to a virtual server by using the command line

At the command prompt, run the following command:

bind lb vserver <name> <serviceName>
Example

```
> bind lb vserver v1 sr1
```

To bind a service to a virtual server by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers.
2. Open a virtual server, and click in the Services pane to bind a service to the virtual server.

Bind the certificate-key pair to the virtual server on the ADC

For more information, see Binding the Certificate-Key Pair to the SSL-Based Virtual Server.

To bind a certificate-key pair to a virtual server by using the command line

At the command prompt, run the following command:

bind ssl vserver <vServerName> -certkeyName <string>

Example

```
> bind ssl vserver v1 -certkeyName key22
Warning: Current certificate replaces the previous binding
```

To bind a certificate-key pair to a virtual server by using the configuration utility

1. Navigate to Traffic Management > Load Balancing > Virtual Servers.
2. Open an SSL virtual server and, in Advanced Settings, click SSL Certificate.
3. Bind a server certificate to the virtual server.

Verify the configuration

To view the configuration by using the command line

At the command prompt, run the following commands:

- show lb vserver <name>
- show ssl vserver <vServerName>

Example

```
> show lb vserver v1
        v1 (192.168.17.252:443) - SSL   Type: ADDRESS
        State: UP
        Last state change was at Wed Oct 29 03:11:11 2014
        Time since last state change: 0 days, 00:01:25.220
        Effective State: UP
        Client Idle Timeout: 180 sec
        Down state flush: ENABLED
        Disable Primary Vserver On Down : DISABLED
        Appflow logging: ENABLED
        No. of Bound Services :  1 (Total)       1 (Active)
        Configured Method: LEASTCONNECTION
        Current Method: Round Robin, Reason: Bound service's state changed to UP
        Mode: IP
        Persistence: NONE
        Vserver IP and Port insertion: OFF
        Push: DISABLED  Push VServer:
        Push Multi Clients: NO
        Push Label Rule: none
        L2Conn: OFF
        Skip Persistency: None
        IcmpResponse: PASSIVE
        RHIstate: PASSIVE
        New Service Startup Request Rate: 0 PER_SECOND, Increment Interval: 0
        Mac mode Retain Vlan: DISABLED
        DBS_LB: DISABLED
        Process Local: DISABLED
        Traffic Domain: 0

1) sr1 (192.168.17.246: 80) - HTTP State: UP    Weight: 1
Done
>

> sh ssl vserver v1
        Advanced SSL configuration for VServer v1:
        DH: DISABLED
        Ephemeral RSA: ENABLED          Refresh Count: 0
        Session Reuse: ENABLED          Timeout: 120 seconds
        Cipher Redirect: DISABLED
        SSLv2 Redirect: DISABLED
        ClearText Port: 0
        Client Auth: DISABLED
        SSL Redirect: DISABLED
        Non FIPS Ciphers: DISABLED
        SNI: DISABLED
        SSLv2: DISABLED  SSLv3: DISABLED  TLSv1.0: ENABLED  TLSv1.1: DISABLED  TLSv1.2: DISA
        Push Encryption Trigger: Always
        Send Close-Notify: YES
        ECC Curve: P_256, P_384, P_224, P_521

1)      CertKey Name: key22       Server Certificate

1)      Cipher Name: DEFAULT
        Description: Predefined Cipher Alias
 Done
```

To view the configuration by using the configuration utility

Navigate to Traffic Management > Load Balancing > Virtual Servers, and double-click an SSL virtual server to open it and view the configuration.

Limitations

- SSL version 3 (SSLv3) is not supported on an MPX appliance but is supported on a VPX virtual appliance. A VPX instance provisioned on an SDX appliance supports SSLv3 only if an SSL chip is not assigned to the instance.
- ECDHE/DHE ciphers and Export ciphers are not supported.
- SSL server key exchange using HSM keys is not supported.
- If you have added or removed keys after you last saved the configuration, you must save the configuration before you perform a warm restart. If you do not save the configuration, there will be a key mismatch between the ADC and the HSM.
- You cannot bind an HSM key to a DTLS virtual server.
- You cannot bind a certificate-key pair that is created by using an HSM key to an SSL service.
- You cannot use the NetScaler configuration utility to enroll the ADC as a client of the HSM or check the status of the HSM from the configuration utility.
- From release 11 build 62.x, SSL renegotiation is supported.
- You cannot sign OCSP requests by using a certificate-key pair that is created by using an HSM key.
- A certificate bundle with HSM keys is not supported.
- An error does not appear if the HSM key and certificate do not match. Therefore, while adding a certificate-key pair, you need to make sure that the HSM key and certificate match.

Appendix

Example

Note: In the following example, the commands are in boldface type.

```
root@ns# ./chkserv
nCipher server running
root@ns# ./nfkminfo
World
 generation  2
 state       0x17a70000 Initialised Usable Recovery PINRecovery !ExistingClient RTC NVRAM FT
 n_modules   1
 hknso       cbec8c0c56c6b5e76b73147ef02d34a661eaa044
 hkm     bbb8d4839da5782be4d092735a7535538834dc91 (type Rijndael)
 hkmwk       1d572201be533ebc89f30fdd8f3fac6ca3395bf0
 hkre        01f21ecf43933ffdd45e74c3883525176c5c439c
 hkra        ac8ec5ee6bce00991bd97adce2091d9739b9b452
 hkmc        cf1b509abaad91995ed202d8f36613fc99433155
 hkp         c20910b2ed1ca62d6a2b0db67052a05f7bbfeb43
 hkrtc       bd811020a7c2f8df435a481c3767a89c2e13bc4f
 hknv        278b8012e48910d518a9ee91cff57233fb0c9093
 hkdsee      12230b0e31e3cec66324c0815f782cfb9249edd5
 hkfto       89dd6250b3d6149bcd15606f4553085e2fd6271a
 hkmnull     0100000000000000000000000000000000000000
 ex.client   none
 k-out-of-n  1/2
 other quora m=1 r=1 p=1 nv=1 rtc=1 dsee=1 fto=1
 createtime  2014-02-28 21:05:32
 nso timeout 10 min
 ciphersuite DLf1024s160mRijndael

Module #1
 generation 2
 state      0x2 Usable
 flags      0x10000 ShareTarget
 n_slots    2
 esn        BD17-C807-58D9
 hkml       70289a6edba00ddc7e3f6d6f5a49edc963e822f2

Module #1 Slot #0 IC 0
 generation    1
 phystype      SmartCard
 slotlistflags 0x2 SupportsAuthentication
 state         0x2 Empty
 flags         0x0
 shareno       0
 shares
 error         OK
No Cardset

Module #1 Slot #1 IC 0
 generation    1
 phystype      SoftToken
 slotlistflags 0x0
 state         0x2 Empty
 flags         0x0
 shareno       0
 shares
 error         OK
No Cardset

No Pre-Loaded Objects

root@ns# ./sigtest
 Hardware module #1 speed index 5792 recommended minimum queue 19
Found 1 module; using 19 jobs
Making 1024-bit RSAPrivate key on module #1;
  using Mech_RSApPKCS1 and PlainTextType_Bignum.
Generated and exported key from module #1.
Imported keys on module #1
 1,       3059 1223.6, 3059 overall
 2,       8698 2989.76, 4349 overall
```

```
3,      14396 4073.06, 4798.67 overall
4,      20091 4721.83, 5022.75 overall
5,      25799 5116.3, 5159.8 overall
6,      31496 5348.58, 5249.33 overall
7,      37192 5487.55, 5313.14 overall
8,      42780 5527.73, 5347.5 overall
9,      45777 4515.44, 5086.33 overall
10,      51457 4981.26, 5145.7 overall
11,      57151 5266.36, 5195.55 overall
12,      62813 5424.61, 5234.42 overall
13,      68496 5527.97, 5268.92 overall
14,      74182 5591.18, 5298.71 overall
15,      79832 5614.71, 5322.13 overall
16,      85518 5643.23, 5344.88 overall
17,      88412 4543.54, 5200.71 overall
18,      94086 4995.72, 5227 overall
19,      99778 5274.23, 5251.47 overall
20,     105469 5440.94, 5273.45 overall
21,     111133 5530.16, 5292.05 overall
22,     116838 5600.1, 5310.82 overall
23,     122522 5633.66, 5327.04 overall
24,     128175 5641.4, 5340.62 overall
25,     131072 4543.64, 5242.88 overall
26,     136762 5002.18, 5260.08 overall
27,     142415 5262.51, 5274.63 overall
28,     148125 5441.51, 5290.18 overall
29,     153816 5541.3, 5304 overall
30,     159414 5563.98, 5313.8 overall
```

Support for SafeNet Network Hardware Security Module

A non-FIPS NetScaler appliance stores the serverâ€™s private key on the hard disk. On a FIPS appliance, the key is stored in a cryptographic module known as a hardware security module (HSM). Storing a key in the HSM protects it from physical and software attacks. In addition, the keys are encrypted with special FIPS approved ciphers.

Only the NetScaler MPX 9700/10500/12500/15500 FIPS appliances support a FIPS card. Support for FIPS is not available on other MPX appliances, or on the SDX and VPX appliances. This limitation is addressed by supporting a SafeNet network HSM on all NetScaler MPX, SDX, and VPX appliances except the MPX 9700/10500/12500/15500 FIPS appliances.

A SafeNet network HSM is designed to protect critical cryptographic keys and to accelerate sensitive cryptographic operations across a wide range of security applications.

Prerequisites

Before you can use a SafeNet network HSM with a NetScaler ADC, make sure that the following prerequisites are met:

- A SafeNet network HSM is installed in the network, ready to use, and accessible to the NetScaler ADC. Tha is, the NetScaler IP (NSIP) address is added as an authorized client on the HSM.
- Licenses are available to support the required number of partitions on the HSM.
- The SafeNet network HSM and the NetScaler ADC can initiate connections with each other through port 1792.
- You are using NetScaler release 11.1.
- The NetScaler appliance does not contain a FIPS Cavium card.

Important

SafeNet network HSMs are not supported on the MPX 9700/10500/12500/15500 FIPS appliances.

Configuring a SafeNet Client on the NetScaler ADC

After you have configured the SafeNet HSM and created the required partitions, you must create clients and assign them to partitions. Begin by configuring the SafeNet clients on the NetScaler ADC and setting up the network trust links (NTLs) between the SafeNet clients and the SafeNet HSM. A sample configuration is given in the Appendix.

1) Create a safenet directory on the NetScaler ADC.

When you load the NetScaler build by using the installns script, the safenet_dirs.tar file is copied into the /var/ directory. If noâ€œ/var/safenet/â€• directory is present, the installns script creates a â€œsafenetâ€• directory in the /var/ directory.

2) Configure the NTLs between SafeNet client (ADC) and HSM.

After the â€œ/var/safenet/â€• directory is created, perform the following tasks on the ADC.

a) Change directory to /var/safenet/config/ and run the â€œsafenet_configâ€• script. At the shell prompt, type:

cd /var/safenet/config

sh safenet_config

This script copies the â€œChrystoki.confâ€• file into the /etc/ directory. It also generates a symbolic link â€œ libCryptoki2_64.soâ€• in the â€œ/usr/lib/â€• directory.

b) Create and transfer a certificate and key between the ADC and the SafeNet HSM.

In order to communicate securely, the ADC and the HSM must exchange certificates. Create a certificate and key on the ADC and then transfer it to the HSM. Copy the HSM certificate to the ADC.

i) Change directory to /var/safenet/safenet/lunaclient/bin.

ii) Create a certificate on the ADC. At the shell prompt, type:

./vtl createCert -n <ip address of NetScaler>

This command also adds the certificate and key path to the â€œ/etc/Chrystoki.confâ€• file.

iii) Copy this certificate to the HSM. At the shell prompt, type:

scp /var/safenet/safenet/lunaclient/cert/client/<ip address of NS>**.pem** <LunaSA_HSM account>@<IP address of Luna SA>

iv) Copy the HSM certificate to the NetScaler ADC. At the shell prompt, type:

scp <HSM account>@<HSM IP>**:server.pem /var/safenet/safenet/lunaclient/server**_<HSM ip>.pem

3) Register the NetScaler ADC as a client and assign it a partition on the SafeNet HSM.

Log on to the HSM and create a client. Enter the NSIP as the client IP. This must be the IP address of the ADC from which you transferred the certificate to the HSM. After the client is successfully registered, assign a partition to it. Run the following commands on the HSM.

a) Use SSH to connect to the SafeNet HSM and enter the password.

b) Register the NetScaler ADC on the SafeNet HSM. The client is created on the HSM. The IP address is the client's IP address. That is, the NSIP address.

At the prompt, type:

client register â€“**client** <client name> -**ip** <netscaler ip>

c) Assign the client a partition from the partition list. To view the available partitions, type:

<luna_sh> **partition list**

Assign a partition from this list. Type:

<lunash:> **client assignPartition** -**client** <Client Name> -**par** <Partition Name>

4) Register the HSM with its certificate on the NetScaler ADC.

On the ADC, change directory to â€œ/var/safenet/safenet/lunaclient/binâ€• and, at the shell prompt, type:

./vtl addserver -**n** <IP addr of HSM> -**c /var/safenet/safenet/lunaclient/server**_<HSM_IP>**.pem**

To remove the HSM that is enrolled on the ADC, type:

./vtl deleteServer -**n** <HSM IP> -**c** <cert path>

To list the HSM servers configured on the ADC, type:

./vtl listServer

5) Verify the network trust links (NTLs) connectivity between the ADC and HSM. At the shell prompt, type:

./vtl verify

If verification fails, review all the steps. Errors are generally due to an incorrect IP address in the client certificates.

6) Save the configuration.

The above steps update the â€œ/etc/Chrystoki.confâ€• configuration file. This file is deleted when the ADC is started. Copy the configuration to the default configuration file, which is used when an ADC is restarted.

At the shell prompt, type:

root@ns# **cp /etc/Chrystoki.conf /var/safenet/config/**

Recommended practice is to run this command every time there is a change to the SafeNet-related configuration.

7) Start the SafeNet gateway process.

At the shell prompt, type:

sh /var/safenet/gateway/start_safenet_gw

8) Configure automatic start of the gateway daemon at boot time.

Create the â€œsafenet_is_enrolledâ€• file, which indicates that SafeNet HSM is configured on this ADC. Whenever the ADC restarts and this file is found, the gateway is automatically started.

At the shell prompt, type:

touch /var/safenet/safenet_is_enrolled

Additional NetScaler Configuration

1) Generate a key on the HSM.
Use third party tools to create keys on the HSM.

2) Add an HSM key on the ADC.

Important! The # character is not supported in a key name. If the key name include this character, the load key operation fails.

To add a Safenet HSM key by using the NetScaler command line

At the command prompt, type:

add ssl hsmkey <KeyName> -**hsmType SAFENET** -**serialNum** <serial #> -**password**

where:

-keyName is the key created on the HSM by using third party tools.

-serialNum is the serial number of the partition on the HSM on which the keys are generated.

-password is the password of the partition on which the keys are present.

To add a Safenet HSM key by using the NetScaler GUI

Navigate to **Traffic Management** > **SSL** > **HSM** and add an HSM key. You must specify the HSM Type as **SAFENET**.

3) Add a certificate-key pair on the ADC. You must first use a third party tool to generate a certificate associated with the key. Then, copy the certificate to the /nsconfig/ssl/ directory on the ADC.

Note: The key must be an HSM key.

To add a certkey pair on the ADC by using the NetScaler command line

At the command prompt, type:

add ssl certkey <CertkeyName> -**cert** <cert name> -**hsmkey** <KeyName>

To add a certkey pair on the ADC by using the NetScaler GUI

1. Navigate to **Traffic Management** > **SSL**.
2. In **Getting Started**, select **Install Certificate (HSM)** and create a certificate-key pair using an HSM key.

4) Create a virtual server and bind the certificate-key pair to this virtual server.

For information about creating a virtual server, see http://docs.citrix.com/en-us/netscaler/11/traffic-management/ssl/config-ssloffloading/config-ssl-vserver.html.

For information about adding a certificate-key pair, see http://docs.citrix.com/en-us/netscaler/11/traffic-management/ssl/config-ssloffloading/add-ssl-certkey.html.

High Availability Setup

You can configure a high availability (HA) setup with a SafeNet HSM configuration in either of the following two ways:

- First, configure a SafeNet HSM on the two nodes, using the same HSM and partition. Then create an HA pair. Finally, add the NetScaler configuration, such as keys, certificate-key pairs, and virtual servers, on the primary node.
- If a SafeNet HSM is already configured on one node with the NetScaler configuration, add a similar configuration on the other node. Copy â€œ/var/safenet/sfgw_ident_fileâ€• from the first node to the other and restart the safenet_gw binary. After the gateway is up and running, add the nodes in an HA setup.

Limitations

1) For any changes to the HSM-related configuration in an existing setup, such as adding or removing an HSM, or creating a high availability setup, you must copy â€œ/etc/Chrystoki.confâ€• to â€œ/var/safenet/configâ€•.

2) After adding, removing, or restarting an HSM, you must restart the â€œ/var/safenet/gateway/safenet_gwâ€• binary. If you donâ€™t restart the gateway binary, the HSM will not serve any traffic after it is added back or after it restarts.

3) To reboot or stop the current â€œ/var/safenet/gateway/safenet_gwâ€• binary, use

- kill â€“SIGTERM <PID>
- kill â€“SIGINT <PID>

Important! Do not use â€œkill â€“9 <PID>â€• or â€œkill -6 <PID>â€•

4) Before removing an existing HSM from the ADC, remove, from the ADC, all the keys and certificate-key pairs that are associated with that HSM. You cannot delete these files from the ADC after you remove the HSM.

5) On a standalone NetScaler appliance, SafeNet HSMs in HA are not supported.

6) EXPORT and DH (ECDHE,DHE,EDH) ciphers are not supported.

7) Update certificate-key pair operation is not supported.

8) When you generate an HSM key on a third-party tool, the private and public key names must be the same. When you add the HSM key on the appliance, provide this name as the key name.

9) The # character is not supported in a key name.

10) Cluster and admin partitions are not supported.

Appendix

Sample commands with their outputs are given below.

run the script

```
root@ns# pwd

/var/safenet/config

root@ns# sh safenet_config
```

Create a certificate

```
root@ns# cd /var/safenet/safenet/lunaclient/bin

root@ns# ./vtl createcert -n 10.102.59.175

Private Key created and written to:
/var/safenet/safenet/lunaclient/cert/client/10.102.59.175Key.pem

Certificate created and written to:
/var/safenet/safenet/lunaclient/cert/client/10.102.59.175.pem
```

Copy the certificate to the HSM

```
root@ns# scp /var/safenet/safenet/lunaclient/cert/client/10.102.59.175.pem
admin@10.217.2.7:

admin@10.217.2.7's password:


10.102.59.175.pem          100%  818     0.8KB/s   00:00
```

Copy the certificate and key from the HSM to the NetScaler appliance

```
root@ns# scp admin@10.217.2.7:server.pem /var/safenet/safenet/lunaclient/server.
2.7.pem

admin@10.217.2.7's password:


server.pem            100% 1164     1.1KB/s   00:01
```

Use SSH to connect to the SafeNet HSM

```
ssh admin@10.217.2.7

Connecting to 10.217.2.7:22...

Connection established.

To escape to local shell, press 'Ctrl+Alt+]'.


Last login: Thu Jun 23 02:20:29 2016 from 10.252.243.11
```

```
Luna SA 5.2.3-1 Command Line Shell - Copyright (c) 2001-2014 SafeNet, Inc. All
rights reserved.


[Safenet1] lunash:>hsm login



  Please enter the HSM Administrators' password:

  > *******


'hsm login' successful.



Command Result : 0 (Success)

[Safenet1] lunash:>
```

Register the NetScaler ADC on the SafeNet HSM

```
[Safenet1] lunash:>client register -client ns175 -ip 10.102.59.175



'client register' successful.



Command Result : 0 (Success)

[Safenet1] lunash:>
```

Assign the client a partition from the partition list

```
[Safenet1] lunash:>client assignPartition -client ns175 -partition p2



'client assignPartition' successful.



Command Result : 0 (Success)

[Safenet1] lunash:>
```

Register the HSM with its certificate on the NetScaler ADC

```
root@ns# ./vtl addserver -n 10.217.2.7 -c /var/safenet/safenet/lunaclient/server.
2.7.pem
```

```
New server 10.217.2.7 successfully added to server list.
```

Verify the network trust links (NTLs) connectivity between the ADC and HSM

```
root@ns# ./vtl verify


The following Luna SA Slots/Partitions were found:


Slot        Serial #                  Label

====      ================           =====

   0              477877010          p2
```

Save the configuration

```
root@ns# cp /etc/Chrystoki.conf /var/safenet/config/
```

Configure automatic start of the gateway daemon at boot time

```
touch /var/safenet/safenet_is_enrolled
```

FAQ

How do I check that the SafeNet process is running?

At the NetScaler shell prompt, type:

ps â€“aux | grep safenet_gw

How do I verify the network trust links (NTLs) connectivity between the ADC and HSM?

After configuring SafeNet, change directory to â€œ/var/safenet/safenet/lunaclient/binâ€• and type:

./vtl verify

Troubleshooting

If the SSL feature does not work as expected after you have configured it, you can use some common tools to access NetScaler resources and diagnose the problem.

Resources for Troubleshooting

Updated: 2013-07-22

For best results, use the following resources to troubleshoot an SSL issue on a NetScaler appliance:

- The relevant ns.log file
- The latest ns.conf file
- The messages file
- The relevant newnslog file
- Trace files
- A copy of the certificate files, if possible
- A copy of the key file, if possible
- The error message, if any

In addition to the above resources, you can use the Wireshark application customized for the NetScaler trace files to expedite troubleshooting.

Troubleshooting SSL Issues

Updated: 2013-07-22

To troubleshoot an SSL issue, proceed as follows:

- Verify that the NetScaler appliance is licensed for SSL Offloading and load balancing.
- Verify that SSL Offloading and load balancing features are enabled on the appliance.
- Verify that the status of the SSL virtual server is not displayed as DOWN.
- Verify that the status of the service bound to the virtual server is not displayed as DOWN.
- Verify that a valid certificate is bound to the virtual server.
- Verify that the service is using an appropriate port, preferably port 443.

SSL FAQs

Basic Questions

HTTPS access to the NetScaler configuration utility fails on a VPX instance. How do I gain access?

A certificate-key pair is required for HTTPS access to the NetScaler configuration utility. On a NetScaler ADC, a certificate-key pair is automatically bound to the internal services. On an MPX or SDX appliance, the default key size is 1024 bytes, and on a VPX instance, the default key size is 512 bytes. However, most browsers today do not accept a key that is less than 1024 bytes. As a result, HTTPS access to the VPX configuration utility is blocked.

Citrix recommends that you install a certificate-key pair of at least 1024 bytes and bind it to the internal service for HTTPS access to the configuration utility or update the ns-server-certificate to 1024 bytes. You can use HTTP access to the configuration utility or the NetScaler command line to install the certificate.

If I add a license to an MPX appliance, the certificate-key pair binding is lost. How do I resolve this problem?

If a license is not present on an MPX appliance when it starts, and you add a license later and restart the appliance, you might lose the certificate binding. You must reinstall the certificate and bind it to the internal service

Citrix recommends that you install an appropriate license before starting the appliance.

What are the various steps involved in setting up a secure channel for an SSL transaction?
Setting up a secure channel for an SSL transaction involves the following steps:

1. The client sends an HTTPS request for a secure channel to the server.
2. After selecting the protocol and cipher, the server sends its certificate to the client.
3. The client checks the authenticity of the server certificate.
4. If any of the checks fail, the client displays the corresponding feedback.
5. If the checks pass or the client decides to continue even if a check fails, the client creates a temporary, disposable key called the *pre-master secret* and encrypts it by using the public key of the server certificate.
6. The server, upon receiving the pre-master secret, decrypts it by using the server's private key and generates the session keys. The client also generates the session keys from the pre-master secret. Thus both client and server now have a common session key, which is used for encryption and decryption of application data.

I understand that SSL is a CPU-intensive process. What is the CPU cost associated with the SSL process?
The following two stages are associated with the SSL process:

- The initial handshake and secure channel setup by using the public and private key technology.
- Bulk data encryption by using the asymmetric key technology.

Both of the preceding stages can affect server performance, and they require intensive CPU processing for of the following reasons:

1. The initial handshake involves public-private key cryptography, which is very CPU intensive because of large key sizes (1024bit, 2048bit, 4096bit).
2. Encryption/decryption of data is also computationally expensive, depending on the amount of data that needs to be encrypted or decrypted.

What are the various entities of an SSL configuration?
An SSL configuration has the following entities:

- Server certificate
- Certificate Authority (CA) certificate
- Cipher suite that specifies the protocols for the following tasks:
  - Initial key exchange
  - Server and client authentication
  - Bulk encryption algorithm
  - Message authentication
- Client authentication
- CRL
- SSL Certificate Key Generation Tool that enables you to create the following files:
  - Certificate request
  - Self signed certificate
  - RSA and DSA keys

DH parameters

I want to use the SSL offloading feature of the Citrix NetScaler appliance. What are the various options for receiving an SSL certificate?

You must receive an SSL certificate before you can configure the SSL setup on the Citrix NetScaler appliance. You can use any of the following methods to receive an SSL certificate:

- Request a certificate from an authorized CA.
- Use the existing server certificate.
- Create a certificate-key pair on the Citrix NetScaler appliance.
  Note: This is a test certificate signed by the test Root-CA generated by the NetScaler. Test certificates signed by this Root-CA are not accepted by browsers. The browser throws a warning message stating that the server's certificate cannot be authenticated.
- For anything other than test purposes, you must provide a valid CA certificate and CA key to sign the server certificate.

What are the minimum requirements for an SSL setup?

The minimum requirements for configuring an SSL setup are as follows:

- Obtain the certificates and keys.
- Create a load balancing SSL virtual server.
- Bind HTTP or SSL services to the SSL virtual server.
- Bind certificate-key pair to the SSL virtual server.

What are the limits for the various components of SSL?

SSL components have the following limits:

- Bit size of SSL certificates: 4096.
- Number of SSL certificates: Depends on the available memory on the appliance.
- Maximum linked intermediate CA SSL certificates: 9 per chain.
- CRL revocations: Depends on the available memory on the appliance.

What are the various steps involved in the end-to-end data encryption on a Citrix NetScaler appliance?

The steps involved in the server-side encryption process on a Citrix NetScaler appliance are as follows:

1. The client connects to the SSL VIP configured on the Citrix NetScaler appliance at the secure site.
2. After receiving the secure request, the appliance decrypts the request, applies layer 4-7 content switching techniques and load balancing policies, and selects the best available backend Web server for the request.
3. The Citrix NetScaler appliance creates an SSL session with the selected server.
4. After establishing the SSL session, the appliance encrypts the client request and sends it to the Web server by using the secure SSL session.
5. When the appliance receives the encrypted response from the server, it decrypts and re-encrypts the data, and sends the data to the client by using the client side SSL session.

The multiplexing technique of the Citrix NetScaler appliance enables the appliance to reuse SSL sessions that have been established with the Web servers. Therefore, the appliance avoids the CPU intensive key exchange, known as *full handshake*. This process reduces the overall number of SSL sessions on the server and maintains end-to-end security.

Certificates and Keys

Can I place the certificate and key files at any location? Is there any recommended location to store these files?

You can store the certificate and key files on the Citrix NetScaler appliance or a local computer. However, Citrix recommends that you store the certificate and key files in the /nsconfig/ssl directory of the Citrix NetScaler appliance. The /etc directory exists in the flash memory of the Citrix NetScaler appliance. This provides portability and facilitates backup and restoration of the certificate files on the appliance.
Note: Make sure that the certificate and the key files are stored in the same directory.

What is the maximum size of the certificate key supported on the Citrix NetScaler appliance?

A Citrix NetScaler appliance running a software release earlier than release 9.0 supports a maximum certificate key size of 2048 bits. Release 9.0 and later support a maximum certificate key size of 4096 bits. This limit is applicable to both RSA and DSA certificates.
An MPX appliance supports certificates from 512-bits up to the following sizes:

- 4096-bit server certificate on the virtual server
- 4096-bit client certificate on the service
- 4096-bit CA certificate (includes intermediate and root certificates)

- 4096-bit certificate on the back end server
- 4096-bit client certificate (if client authentication is enabled on the virtual server)

A virtual appliance supports certificates from 512-bits up to the following sizes:

- 4096-bit server certificate on the virtual server
- 4096-bit client certificate on the service
- 4096-bit CA certificate (includes intermediate and root certificates)
- 2048-bit certificate on the back end server
- 2048-bit client certificate (if client authentication is enabled on the virtual server)

What is the maximum size of the DH parameter supported on the Citrix NetScaler appliance?
The Citrix NetScaler appliance supports a DH parameter of maximum 2048 bits.

What is the maximum certificate-chain length, that is, the maximum number of certificates in a chain, supported on a Citrix NetScaler appliance?
A Citrix NetScaler appliance can send a maximum of 10 certificates in a chain when sending a server certificate message. A chain of the maximum length includes the server certificate and nine intermediate CA certificates.

What are the various certificate and key formats supported on the Citrix NetScaler appliance?
The Citrix NetScaler appliance supports the following certificate and key formats:

- Privacy Enhanced Mail (PEM)
- Distinguished Encoding Rule (DER)

Is there a limit for the number of certificates and keys that I can install on the Citrix NetScaler appliance?
No. The number of certificates and keys that can be installed is limited only by the available memory on the Citrix NetScaler appliance.

I have saved the certificate and key files on the local computer. I want to transfer these files to the Citrix NetScaler appliance by using the FTP protocol. Is there any preferred mode for transfering these files to the Citrix NetScaler appliance?
Yes. If using the FTP protocol, you should use binary mode to transfer the certificate and key files to the Citrix NetScaler appliance.
Note: By default, FTP is disabled. Citrix recommends using the SCP protocol for transferring certificate and key files. The configuration utility implicitly uses SCP to connect to the appliance.

What is the default directory path for the certificate and key?
The default directory path for the certificate and key is /nsconfig/ssl.

When adding a certificate and key pair, what happens if I do not specify an absolute path to the certificate and key files?
When adding a certificate and key pair, if you do not specify an absolute path to the certificate and key files, the Citrix NetScaler appliance searches the default directory, /nsconfig/ssl, for the specified files and attempts to load them to the kernel. For example, if the cert1024.pem and rsa1024.pem files are available in the /nsconfig/ssl directory of the appliance, both of the following commands are successful:

```
add ssl certKey cert1 -cert cert1204.pem -key rsa1024.pem
```

```
add ssl certKey cert1 -cert /nsconfig/ssl/cert1204.pem -key /nsconfig/ssl/rsa1024.
pem
```

I have configured a high availability setup. I want to implement the SSL feature on the setup. How should I handle the certificate and key files in a high availability setup?
In a high availability setup, you must store the certificate and key files on both the primary and the secondary Citrix NetScaler appliance. The directory path for the certificate and key files must be the same on both appliances before you add an SSL certificate-key pair on the primary appliance.

Ciphers

What is a NULL-Cipher?
Ciphers with no encryption are known as NULL-Ciphers. For example, NULL-MD5 is a NULL-Cipher.

Are the NULL-Ciphers enabled by default for an SSL VIP or an SSL service?
No. NULL-Ciphers are not enabled by default for an SSL VIP or an SSL service.

What is the procedure to remove NULL-Ciphers?
To remove the NULL-Ciphers from an SSL VIP, run the following command:

bind ssl cipher <SSL_VIP> REM NULL

To remove the NULL-Ciphers from an SSL Service, run the following command:

```
bind ssl cipher <SSL_Service> REM NULL -service
```

What are the various cipher aliases supported on the Citrix NetScaler appliance?
The Citrix NetScaler appliance supports the following cipher aliases:

1. Alias Name: ALL

   Description: All NetScaler-supported ciphers, excluding NULL ciphers
2. Alias Name: DEFAULT

   Description: Default cipher list with encryption strength >= 128bit
3. Alias Name: kRSA

   Description: Ciphers with RSA key exchange algorithm
4. Alias Name: kEDH

   Description: Ciphers with Ephemeral-DH key exchange algorithm
5. Alias Name: DH

   Description: Ciphers with DH key exchange algorithm
6. Alias Name: EDH

   Description: Ciphers with DH key exchange algorithm and authentication algorithm
7. Alias Name: aRSA

   Description: Ciphers with RSA authentication algorithm
8. Alias Name: aDSS

   Description: Ciphers with DSS authentication algorithm
9. Alias Name: aNULL

   Description: Ciphers with NULL authentication algorithm
10. Alias Name: DSS

    Description: Ciphers with DSS authentication algorithm
11. Alias Name: DES

    Description: Ciphers with DES encryption algorithm
12. Alias Name: 3DES

    Description: Ciphers with 3DES encryption algorithm
13. Alias Name: RC4

    Description: Ciphers with RC4 encryption algorithm
14. Alias Name: RC2

    Description: Ciphers with RC2 encryption algorithm
15. Alias Name: eNULL

    Description: Ciphers with NULL encryption algorithm
16. Alias Name: MD5

    Description: Ciphers with MD5 message authentication code (MAC) algorithm
17. Alias Name: SHA1

    Description: Ciphers with SHA-1 MAC algorithm
18. Alias Name: SHA

    Description: Ciphers with SHA MAC algorithm
19. Alias Name: NULL

    Description: Ciphers with NULL encryption algorithm
20. Alias Name: RSA

Description: Ciphers with RSA key exchange algorithm and authentication algorithm

21. Alias Name: ADH

    Description: Ciphers with DH key exchange algorithm, and NULL authentication algorithm
22. Alias Name: SSLv2

    Description: SSLv2 protocol ciphers
23. Alias Name: SSLv3

    Description: SSLv3 protocol ciphers
24. Alias Name: TLSv1

    Description: SSLv3/TLSv1 protocol ciphers
25. Alias Name: TLSv1_ONLY

    Description: TLSv1 protocol ciphers
26. Alias Name: EXP

    Description: Export ciphers
27. Alias Name: EXPORT

    Description: Export ciphers
28. Alias Name: EXPORT40

    Description: Export ciphers with 40-bit encryption
29. Alias Name: EXPORT56

    Description: Export ciphers with 56-bit encryption
30. Alias Name: LOW

    Description: Low strength ciphers (56-bit encryption)
31. Alias Name: MEDIUM

    Description: Medium strength ciphers (128-bit encryption)
32. Alias Name: HIGH

    Description: High strength ciphers (168-bit encryption)
33. Alias Name: AES

    Description: AES ciphers
34. Alias Name: FIPS

    Description: FIPS-approved ciphers
35. Alias Name: ECDHE

    Description: Elliptic Curve Ephemeral DH Ciphers

What is the command to display all the predefined ciphers of the Citrix NetScaler appliance?

To display all the predefined ciphers of the Citrix NetScaler appliance, at the NetScaler command line, type:

show ssl cipher

What is the command to display the details of an individual cipher of the Citrix NetScaler appliance?

To display the details of an individual cipher of the Citrix NetScaler appliance, at the NetScaler command line, type:

show ssl cipher <Cipher_Name/Cipher_Alias_Name/Cipher_Group_Name>

Example:

```
 > show cipher SSL3-RC4-SHA
 1) Cipher Name: SSL3-RC4-SHA
 Description: SSLv3 Kx=RSA Au=RSA Enc=RC4(128)
Mac=SHA1
 Done
```

What is the significance of adding the predefined ciphers of the Citrix NetScaler appliance?

Adding the predefined ciphers of the Citrix NetScaler appliance causes the NULL-Ciphers to get added to an SSL VIP or an SSL service.

Certificates

Is the distinguished name in a client certificate available for the length of the user session?

Yes. You can access the distinguished name of the client certificate in subsequent requests during the length of the user session, that is even after the SSL handshake is complete and the certificate is not sent again by the browser. To do this, use a variable and an assignment as detailed in the following sample configuration:

Example:

```
add ns variable v2 -type "text(100)"
add ns assignment a1 -variable "$v2" -set      "CLIENT.SSL.CLIENT_CERT.SUBJECT.TYPECAST_NVLIST_T(\'=\',\'/\').
VALUE(\"CN\")"

add rewrite action act1 insert_http_header subject "$v2"  // example: to insert the distinguished name in the header
add rewrite policy pol1 true a1
add rewrite policy pol2 true act1
bind rewrite global pol1 1 next -type RES_DEFAULT
bind rewrite global pol2 2 next -type RES_DEFAULT
set rewrite param -undefAction RESET
```

Why do I need to bind the server certificate?

Binding the server certificates is the basic requirement for enabling the SSL configuration to process SSL transactions

To bind the server certificate to an SSL VIP, at the NetScaler command line, type:

```
bind ssl vserver <vServerName> -certkeyName <cert_name>
```

To bind the server certificate to an SSL service, at the NetScaler command line, type:

```
bind ssl service <serviceName> -certkeyName <cert_name>
```

How many certificates can I bind to an SSL VIP or an SSL service?

On a NetScaler virtual appliance, you can bind a maximum of two certificates to an SSL VIP or an SSL service, one each of type RSA and type DSA. On a NetScaler MPX or MPX-FIPS appliance, if SNI is enabled, you can bind multiple server certificates of type RSA. If SNI is disabled, you can bind a maximum of one certificate of type RSA. Note: DSA certificates are not supported on MPX or MPX-FIPS platforms.

Does SNI support Subject Alternative Name (SAN) certificates?

No. On a NetScaler appliance, SNI is not supported with a SAN extension certificate.

What happens if I unbind or overwrite a server certificate?

When you unbind or overwrite a server certificate, all the connections and SSL sessions created by using the existing certificate are terminated. When you overwrite an existing certificate, the following message appears:

```
ERROR:

Warning: Current certificate replaces the previous binding.
```

How do I install an intermediate certificate on Citrix NetScaler and link to a server certificate?

See the article at http://support.citrix.com/article/ctx114146 for information about installing an intermediate certificate.

Why am I am getting a "resource already exists" error when I try to install a certificate on the Citrix NetScaler?

See the article at http://support.citrix.com/article/CTX117284 for instructions for resolving the "resource already exists" error.

I want to create a server certificate on a Citrix NetScaler appliance to test and evaluate the product. What is the procedure to create a server certificate?

Perform the following procedure to create a test certificate.
Note: A certificate created with this procedure cannot be used to authenticate all the users and browsers. After using the certificate for testing, you should obtain a server certificate signed by an authorized Root CA.
To create a self-signed server certificate:

1. To create a Root CA certificate, at the NetScaler command line, type:

   ```
   create ssl rsakey /nsconfig/ssl/test-ca.key 1024
   ```

create ssl certreq /nsconfig/ssl/test-ca.csr -keyfile /nsconfig/ssl/test-ca.key

Enter the required information when prompted, and then type the following command:

create ssl cert /nsconfig/ssl/test-ca.cer /nsconfig/ssl/test-ca.csr ROOT_CERT -keyfile /nsconfig/ssl/test-ca.key

2. Perform the following procedure to create a server certificate and sign it with the root CA certificate that you just created
   a. To create the request and the key, at the NetScaler command line, type:

      create ssl rsakey /nsconfig/ssl/test-server.key 1024

      create ssl certreq /nsconfig/ssl/test-server.csr -keyfile /nsconfig/ssl/test-server.key
   b. Enter the required information when prompted.
   c. To create a serial-number file, at the NetScaler command line, type:

```
 shell
 # echo '01' >
/nsconfig/ssl/serial.txt
 # exit
```

3. To create a server certificate signed by the root CA certificate created in step 1, at the NetScaler command line, type:

   create ssl cert /nsconfig/ssl/test-server.cer /nsconfig/ssl/test-server.csr SRVR_CERT -CAcert /nsconfig/ssl/test-ca.cer -CAkey /nsconfig/ssl/test-ca.key -CAserial /nsconfig/ssl/serial.txt
4. To create a Citrix NetScaler certkey, which is the in-memory object that holds the server certificate information for SSL handshakes and bulk encryption, at the NetScaler command line, type:

   add ssl certkey test-certkey -cert /nsconfig/ssl/test-server.cer -key /nsconfig/ssl/test-server.key
5. To bind the certkey object to the SSL virtual server, at the NetScaler command line, type:

   bind ssl vserver <vServerName> -certkeyName <cert_name>

I have received a Citrix NetScaler appliance on which Citrix NetScaler software release 9.0 is installed. I have noticed an additional license file on the appliance. Is there any change in the licensing policy starting with Citrix NetScaler software release 9.0?

Yes. Starting with Citrix NetScaler software release 9.0, the appliance might not have a single license file. The numbe of license files depends on the Citrix NetScaler software release edition. For example, if you have installed the Enterprise edition, you might need additional license files for the full functionality of the various features. However, if you have installed the Platinum edition, the appliance has only one license file.

How do I export the certificate from Internet Information Service (IIS)?

There are many ways to do this, but by using the following method the appropriate certificate and private key for the Web site are exported. This procedure **must** be performed on the actual IIS server.

1. Open the Internet Information Services (IIS) Manager administration tool.
2. Expand the Web Sites node and locate the SSL-enabled Web site that you want to serve through the Citrix NetScaler.
3. Right-click this Web site and click Properties.
4. Click the Directory Security tab and, in the Secure Communications section of the window, select the View Certificate box.
5. Click the Details tab, and then click Copy to File.
6. On the Welcome to the Certificate Export Wizard page, click Next.
7. Select Yes, export the private key and click Next.
   Note: The private key MUST be exported for SSL Offload to work on the Citrix NetScaler
8. Make sure that the Personal Information Exchange -PKCS #12 radio button is selected, and select *only* the Include all certificates in the certification path if possible check box. Click Next.
9. Enter a password and click Next.
10. Enter a file name and location, and then click Next. Give the file an extension of .PFX.
11. Click Finish.

How do I convert the PKCS#12 certificate and install it on the Citrix NetScaler?

1. Move the exported .PFX certificate file to a location from where it may be copied to the Citrix NetScaler (that is to a machine that permits SSH access to the management interface of a Citrix NetScaler appliance). Copy the certificate to the appliance by using a secure copy utility such as SCP.
2. Access the BSD shell and convert the certificate (for example, cert.PFX) to .PEM format:

```
root@ns# openssl pkcs12 -in cert.PFX -out cert.PEM
```

3. To make sure that the converted certificate is in correct x509 format, verify that the following command produces no error:

   ```
   root@ns# openssl x509 -in cert.PEM -text
   ```

4. Verify that the certificate file contains a private key. Begin by issuing the following command:

   ```
   root@ns# cat cert.PEM
   ```

   Verify that the output file includes an RSA PRIVATE KEY section.

   ```
   -----BEGIN RSA PRIVATE KEY-----
    Mkm^s9KMs9023pz/s...
    -----END RSA PRIVATE KEY-----
   ```

   The following is another example of an RSA PRIVATE KEY section:

   ```
    Bag Attributes
   1.3.6.1.4.1.311.17.2: <No Values>
   localKeyID: 01 00 00 00
   Microsoft CSP Name: Microsoft RSA SChannel Cryptographic
   Provider
   friendlyName:
   4b9cef4cc8c9b849ff5c662fd3e0ef7e_76267e3e-6183-4d45-886e-6e067297b38f

   Key Attributes
   X509v3 Key Usage: 10
   -----BEGIN RSA PRIVATE KEY-----
   Proc-Type: 4,ENCRYPTED
   DEK-Info: DES-EDE3-CBC,43E7ACA5F4423968
   pZJ2SfsSVqMbRRf6ug37Clua5gY0Wld4frPIxFXyJquUHr31dilW5ta3hbIaQ+Rg

   ... (more random characters)
   v8dMugeRplkaH2Uwt/mWBk4t71Yv7GeHmcmjafK8H8iW80ooPO3D/ENV8X4U/tlh

   5eU6ky3WYZ1BTy6thxxLlwAullynVXZEflNLxq1oX+ZYl6djgjE3qg==
   -----END RSA PRIVATE KEY-----
   ```

   The following is a SERVER CERTIFICATE section:

   ```
   Bag Attributes
   localKeyID: 01 00 00 00
   friendlyName: AG Certificate
   subject=/C=AU/ST=NSW/L=Wanniassa/O=Dave Mother
   Asiapacific/OU=Support/CN=davemother.food.lan
   issuer=/DC=lan/DC=food/CN=hotdog
   -----BEGIN CERTIFICATE-----
   MIIFiTCCBHGgAwIBAgIKCGryDgAAAAAAHzANBgkqhkiG9w0BAQUFADA8MRMwEQYK

   ... (more random characters) 5pLDWYVHhLkA1pSxvFjNJHRSIydWHc5ltGyKqIUcBezVaXyel94p

   MY2ovQyQZM8gGe3+lGFum0VHbv/y/gB9HhFesog=
   -----END CERTIFICATE-----
   ```

   The following is an INTERMEDIATE CA CERTIFICATE section:

   ```
   Bag Attributes: <Empty Attributes>
   subject=/DC=lan/DC=food/CN=hotdog
   issuer=/DC=lan/DC=food/CN=hotdog
   -----BEGIN CERTIFICATE-----
   MIIESDCCAzCgAwIBAgIQah20fCRYTY9LRXYMIRaKGjANBgkqhkiG9w0BAQUFADA8

   ... (more random characters) Nt0nksawDnbKo86rQcNnY5xUs7c7pj2zxj/IOsgNHUp5W6dDI9pQ

   -----END CERTIFICATE-----
   ```

   Further Intermediate CA certificates may follow, depending on the certification path of the exported certificate.

5. Open the .PEM file in a text editor
6. Locate the first line of the .PEM file and the first instance of the following line, and copy those two lines and all the lines between them:

   ```
   -----END CERTIFICATE-----
   ```

   Note: Make sure that last copied line is the first `-----END CERTIFICATE-----` line in the .PEM file.
7. Paste the copied lines into a new file. Call the new file something intuitive, such as `cert-key.pem`. This is the certificate-key pair for the server hosting the HTTPS service. This file should contain both the section labeled `RSA PRIVATE KEY` and the section labeled `SERVER CERTIFICATE` in the example above.
   Note: The certificate-key pair file contains the private key and must therefore be kept secure.
8. Locate any subsequent sections beginning with `-----BEGIN CERTIFICATE-----` and ending with `---END CERTIFICATE-----`, and copy each such section to a separate new file.

   These sections correspond to certificates of trusted CAs that have been included in the certification path. These sections should be copied and pasted into new individual files for these certificates. For example, the INTERMEDIATE CA CERTIFICATE section of the example above should be copied and pasted into a new file).

   For multiple intermediate CA certificates in the original file, create new files for each intermediate CA certificate in the order in which they appear in the file. Keep track (using appropriate filenames) of the order in which the certificates appear, as they need to be linked together in the correct order in a later step.
9. Copy the certificate-key file (`cert-key.pem`) and any additional CA certificate files into the /nsconfig/ssl directory on the Citrix NetScaler.
10. Exit the BSD shell and access the Citrix NetScaler prompt.
11. Follow the steps in "Install the certificate-key files on the appliance" to install the key/certificate once uploaded on the device.

How do I convert the PKCS#7 certificate and install it on the NetScaler appliance?

You can use OpenSSL to convert a PKCS #7 Certificate to a format recognizable by the NetScaler appliance. The procedure is identical to the procedure for PKCS #12 certificates, except that you invoke OpenSSL with different parameters. The steps for converting PKCS #7 certificates are as follows:

1. Copy the certificate to the appliance by using a secure copy utility, such as SCP.
2. Convert the certificate (for example, cert.P7B) to PEM format:

   ```
   > openssl pkcs7 -inform DER -in cert.p7b -print_certs -text -out cert.pem
   ```
3. Follow steps 3 through 7 as described in the answer to Q32 for PKCS #12 certificates.

Note: Before loading the converted PKCS #7 certificate to the appliance, be sure to verify that it contains a private key, exactly as described in step 3 for the PKCS #12 procedure. PKCS #7 certificates, particularly those exported from IIS, do not typically contain a private key.

When I bind a cipher to a virtual server or service by using the bind cipher command, I see the error message "Command deprecated."

The command for binding a cipher to a virtual server or service has changed.

Use the bind ssl vserver <vsername> -ciphername <ciphername> command to bind an SSL cipher to an SSL virtual server.

Use the bind ssl service <serviceName> -ciphername <ciphername> command to bind an SSL cipher to an SSL service.

Note: New ciphers and cipher groups are added to the existing list and not replaced.

Why can't I create a new cipher group and bind ciphers to it by using the add cipher command?

The add cipher command functionality has changed in release 10. The command only creates a cipher group. To add ciphers to the group, use the bind cipher command.

OpenSSL

How do I install the OpenSSL toolkit?

See the article at http://support.citrix.com/article/ctx106627.

How do I use OpenSSL to convert certificates between PEM and DER?

To use OpenSSL, you must have a working installation of the OpenSSL software and be able to execute Openssl from the command line.

x509 certificates and RSA keys can be stored in a number of different formats.

Two common formats are DER (a binary format used primarily by Java and Macintosh platforms) and PEM (a base64 representation of DER with header and footer information, which is used primarily by UNIX and Linux platforms). There is also an obsolete NET (Netscape server) format that was used by earlier versions of IIS (up to and including 4.0) and various other less common formats that are not covered in this article.

A key and the corresponding certificate, as well as the root and any intermediate certificates, can also be stored in a single PKCS#12 (.P12, .PFX) file.

Procedure

Use the Openssl command to convert between formats as follows:

1. To convert a certificate from PEM to DER:

   ```
   x509 -in input.crt -inform PEM -out output.crt -outform DER
   ```

2. To convert a certificate from DER to PEM:

   ```
   x509 -in input.crt -inform DER -out output.crt -outform PEM
   ```

3. To convert a key from PEM to DER:

   ```
   rsa -in input.key -inform PEM -out output.key -outform DER
   ```

4. To convert a key from DER to PEM:

   ```
   rsa -in input.key -inform DER -out output.key -outform PEM
   ```

Note: If the key you are importing is encrypted with a supported symmetric cipher, you are prompted to enter the pass phrase.
Note: To convert a key to or from the obsolete NET (Netscape server) format, substitute NET for PEM or DER as appropriate. The stored key is encrypted in a weak unsalted RC4 symmetric cipher, so a pass-phrase will be requested. A blank pass-phrase is acceptable.

System Limits

What are the important numbers to remember?

1. Create Certificate Request:
   - Request File Name: Maximum 63 characters
   - Key File Name: Maximum 63 characters
   - PEM Passphrase (For Encrypted Key): Maximum 31 characters
   - Common Name: Maximum 63 characters
   - City: Maximum 127 characters
   - Organization Name: Maximum 63 characters
   - State/Province Name: Maximum 63 characters
   - Email Address: Maximum 39 Characters
   - Organization Unit: Maximum 63 characters
   - Challenge Password: Maximum 20 characters
   - Company Name: Maximum 127 characters
2. Create Certificate:
   - Certificate File Name: Maximum 63 characters
   - Certificate Request File Name: Maximum 63 characters
   - Key File Name: Maximum 63 characters
   - PEM Passphrase: Maximum 31 characters
   - Validity Period: Maximum 3650 days
   - CA Certificate File Name: Maximum 63 characters
   - CA Key File Name: Maximum 63 characters
   - PEM Passphrase: Maximum 31 characters
   - CA Serial Number File: Maximum 63 characters
3. Create and Install a Server Test Certificate:
   - Certificate File Name: Maximum 31 characters
   - Fully Qualified Domain Name: Maximum 63 characters
4. Create Diffie-Hellman (DH) key:
   - DH Filename (with path): Maximum 63 characters
   - DH Parameter Size: Maximum 2048 bits

5. Import PKCS12 key:
    - Output File Name: Maximum 63 characters
    - PKCS12 File Name: Maximum 63 characters
    - Import Password: Maximum 31 characters
    - PEM Passphrase: Maximum 31 characters
    - Verify PEM Passphrase: Maximum 31 characters
6. Export PKCS12
    - PKCS12 File Name: Maximum 63 characters
    - Certificate File Name: Maximum 63 characters
    - Key File Name: Maximum 63 characters
    - Export Password: Maximum 31 characters
    - PEM Passphrase: Maximum 31 characters
7. CRL Management:
    - CA Certificate File Name: Maximum 63 characters
    - CA Key File Name: Maximum 63 characters
    - CA Key File Password: Maximum 31 characters
    - Index File Name: Maximum 63 characters
    - Certificate File Name: Maximum 63 characters
8. Create RSA Key:
    - Key Filename: Maximum 63 characters
    - Key Size: Maximum 4096 bits
    - PEM Passphrase: Maximum 31 characters
    - Verify Passphrase: Maximum 31 characters
9. Create DSA Key:
    - Key Filename: Maximum 63 characters
    - Key Size: Maximum 4096 bits
    - PEM Passphrase: Maximum 31 characters
    - Verify Passphrase: Maximum 31 characters
10. Change advanced SSL settings:
    - Maximum CRL memory size: Maximum 1024 Mbytes
    - Encryption trigger timeout (10 mS ticks): Maximum 200
    - Encryption trigger packet count: Maximum 50
    - OCSP cache size: Maximum 512 Mbytes
11. Install Certificate:
    - Certificate-Key pair Name: Maximum 31 characters
    - Certificate File Name: Maximum 63 characters
    - Private Key File Name: Maximum 63 characters
    - Password: Maximum 31 characters
    - Notification Period: Maximum 100
12. Create Cipher Group:
    - Cipher Group Name: Maximum 39 characters
13. Create CRL:
    - CRL Name: Maximum 31 characters
    - CRL File: Maximum 63 characters
    - URL: Maximum 127 characters
    - Base DN: Maximum 127 characters
    - Bind DN: Maximum 127 characters
    - Password: Maximum 31 characters
    - Day(s): Maximum 31
14. Create SSL Policy:
    - Name: Maximum 127 characters
15. Create SSL Action:
    - Name: Maximum 127 characters
16. Create OCSP Responder:
    - Name: Maximum 32 characters
    - URL: Maximum 128 characters
    - Batching Depth: Maximum 8
    - Batching Delay: Maximum 10000
    - Produced At Time Skew: Maximum 86400
    - Request Time-out: Maximum120000
17. Create Virtual Server:
    - Name: Maximum 127 characters
    - Redirect URL: Maximum 127 characters
    - Client Time-out: Maximum 31536000 secs

18. Create Service:
    - Name: Maximum 127 characters
    - Idle Time-out (secs):

      Client: Maximum 31536000

      Server: Maximum 31536000
19. Create Service Group:
    - Service Group Name: Maximum 127 characters
    - Server ID: Maximum 4294967295
    - Idle Time-out (secs):

      Client: Maximum value 31536000

      Server: Maximum 31536000
20. Create Monitor:
    - Name: Maximum 31 characters
21. Create Server:
    - Server Name: Maximum 127 characters
    - Domain Name: Maximum 255 characters
    - Resolve Retry: Maximum 20939 secs