CHAPTER 47

Configuring Cisco Unified Presence

This chapter describes how to configure the adaptive security appliance for Cisco Unified Presence.

This chapter includes the following sections:

Information About Cisco Unified Presence

This section includes the following topics:

Architecture for Cisco Unified Presence for SIP Federation Deployments

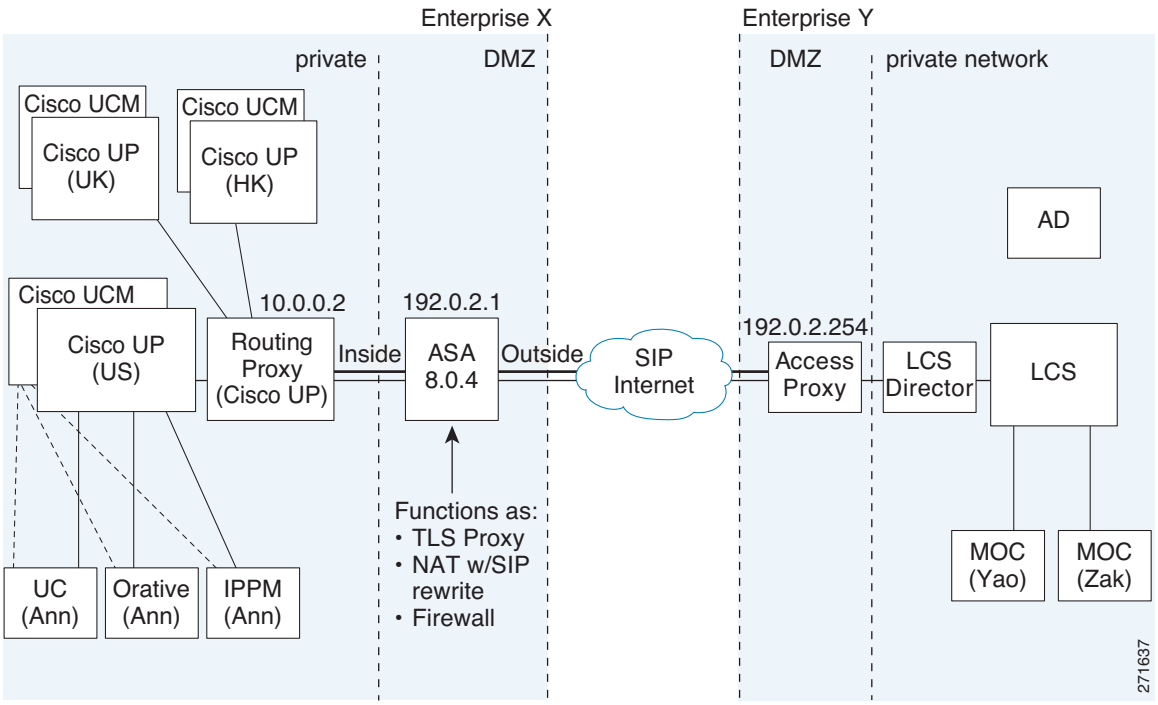
Figure 47-1 depicts a Cisco Unified Presence/LCS Federation scenario with the adaptive security appliance as the presence federation proxy (implemented as a TLS proxy). The two entities with a TLS connection are the “Routing Proxy” (a dedicated Cisco UP) in Enterprise X and the Microsoft Access Proxy in Enterprise Y. However, the deployment is not limited to this scenario. Any Cisco UP or Cisco UP cluster could be deployed on the left side of the adaptive security appliance; the remote entity could be any server (an LCS, an OCS, or another Cisco UP).

The following architecture is generic for two servers using SIP (or other adaptive security appliance inspected protocols) with a TLS connection.

Entity X: Cisco UP/Routing Proxy in Enterprise X

Entity Y: Microsoft Access Proxy/Edge server for LCS/OCS in Enterprise Y

Figure 47-1 Typical Cisco Unified Presence/LCS Federation Scenario



In the above architecture, the adaptive security appliance functions as a firewall, NAT, and TLS proxy, which is the recommended architecture. However, the adaptive security appliance can also function as NAT and the TLS proxy alone, working with an existing firewall.

Either server can initiate the TLS handshake (unlike IP Telephony or Cisco Unified Mobility, where only the clients initiate the TLS handshake). There are by-directional TLS proxy rules and configuration. Each enterprise can have an adaptive security appliance as the TLS proxy.

In Figure 47-1, NAT or PAT can be used to hide the private address of Entity X. In this situation, static NAT or PAT must be configured for foreign server (Entity Y) initiated connections or the TLS handshake (inbound). Typically, the public port should be 5061. The following static PAT command is required for the Cisco UP that accepts inbound connections:

```
hostname(config)# object network obj-10.0.0.2-01
hostname(config-network-object)# host 10.0.0.2
hostname(config-network-object)# nat (inside,outside) static 192.0.2.1 service tcp 5061
5061
```

The following static PAT must be configured for each Cisco UP that could initiate a connection (by sending SIP SUBSCRIBE) to the foreign server.

For Cisco UP with the address 10.0.0.2, enter the following command:

```
hostname(config)# object network obj-10.0.0.2-02
hostname(config-network-object)# host 10.0.0.2
hostname(config-network-object)# nat (inside,outside) static 192.0.2.1 service tcp 5062
5062
hostname(config)# object network obj-10.0.0.2-03
hostname(config-network-object)# host 10.0.0.2
hostname(config-network-object)# nat (inside,outside) static 192.0.2.1 service udp 5070
5070
hostname(config)# object network obj-10.0.0.2-04
hostname(config-network-object)# host 10.0.0.2
```

```
hostname(config-network-object)# nat (inside,outside) static 192.0.2.1 service tcp 5060
5060
```

For another Cisco UP with the address 10.0.0.3, you must use a different set of PAT ports, such as 45062 or 45070:

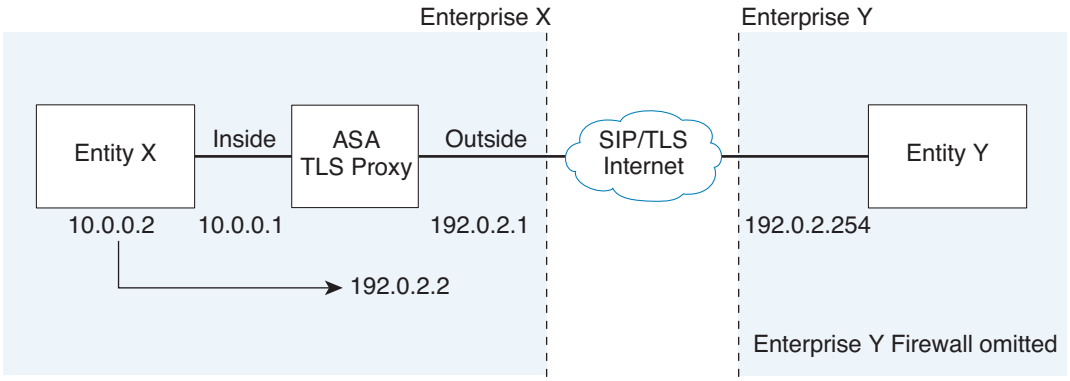
```
hostname(config)# object network obj-10.0.0.3-01
hostname(config-network-object)# host 10.0.0.3
hostname(config-network-object)# nat (inside,outside) static 192.0.2.1 service tcp 5061
45061
hostname(config)# object network obj-10.0.0.3-02
hostname(config-network-object)# host 10.0.0.3
hostname(config-network-object)# nat (inside,outside) static 192.0.2.1 service tcp 5062
45062
hostname(config)# object network obj-10.0.0.3-03
hostname(config-network-object)# host 10.0.0.3
hostname(config-network-object)# nat (inside,outside) static 192.0.2.1 service udp 5070
5070
hostname(config)# object network obj-10.0.0.2-03
hostname(config-network-object)# host 10.0.0.2
hostname(config-network-object)# nat (inside,outside) static 192.0.2.1 service tcp 5070
45070
hostname(config)# object network obj-10.0.0.3-04
hostname(config-network-object)# host 10.0.0.3
hostname(config-network-object)# nat (inside,outside) static 192.0.2.1 service tcp 5060
45060
```

Dynamic NAT or PAT can be used for the rest of the outbound connections or the TLS handshake. The adaptive security appliance SIP inspection engine takes care of the necessary translation (fixup).

```
hostname(config)# object network obj-0.0.0.0-01
hostname(config-network-object)# subnet 0.0.0.0 0.0.0.0
hostname(config-network-object)# nat (inside,outside) dynamic 192.0.2.1
```

Figure 47-2 illustrates an abstracted scenario with Entity X connected to Entity Y through the presence federation proxy on the adaptive security appliance. The proxy is in the same administrative domain as Entity X. Entity Y could have another adaptive security appliance as the proxy but this is omitted for simplicity.

Figure 47-2 Abstracted Presence Federation Proxy Scenario between Two Server Entities



For the Entity X domain name to be resolved correctly when the adaptive security appliance holds its credential, the adaptive security appliance could be configured to perform NAT for Entity X, and the domain name is resolved as the Entity X public address for which the adaptive security appliance provides proxy service.

For further information about configuring Cisco Unified Presence Federation for SIP Federation, see the Integration Guide for Configuring Cisco Unified Presence for Interdomain Federation.:

http://www.cisco.com/en/US/products/ps6837/products_installation_and_configuration_guides_list.html

Trust Relationship in the Presence Federation

Within an enterprise, setting up a trust relationship is achievable by using self-signed certificates or you can set it up on an internal CA.

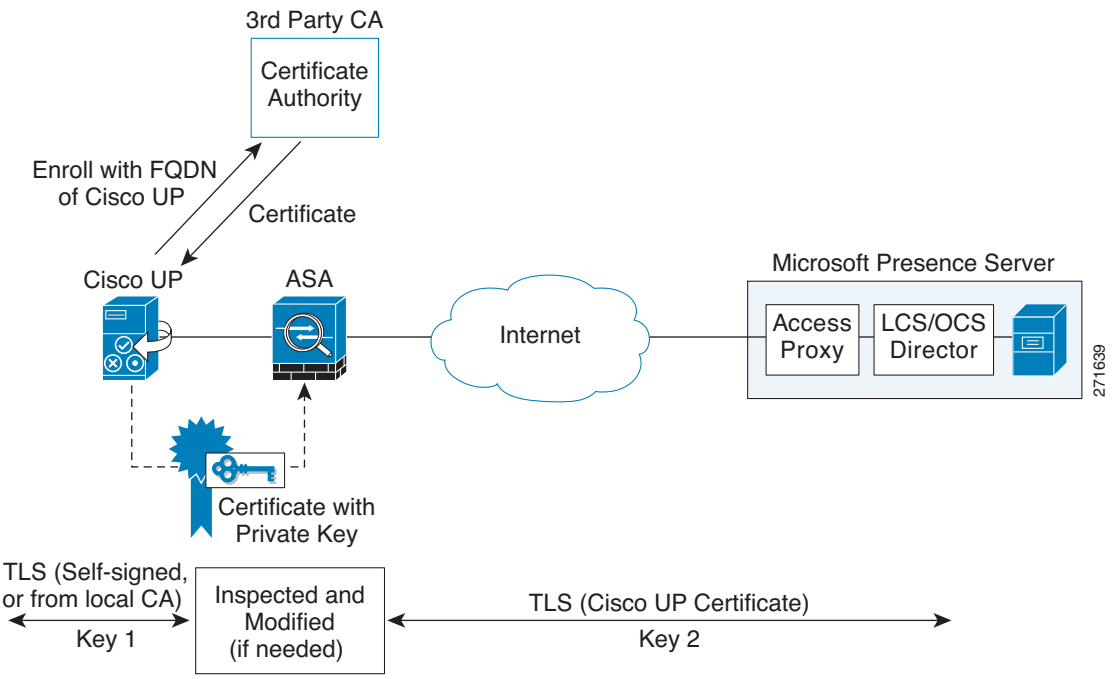
Establishing a trust relationship cross enterprises or across administrative domains is key for federation. Cross enterprises you must use a trusted third-party CA (such as, VeriSign). The adaptive security appliance obtains a certificate with the FQDN of the Cisco UP (certificate impersonation).

For the TLS handshake, the two entities could validate the peer certificate via a certificate chain to trusted third-party certificate authorities. Both entities enroll with the CAs. The adaptive security appliance as the TLS proxy must be trusted by both entities. The adaptive security appliance is always associated with one of the enterprises. Within that enterprise (Enterprise X in Figure 47-1), the entity and the adaptive security appliance could authenticate each other via a local CA, or by using self-signed certificates.

To establish a trusted relationship between the adaptive security appliance and the remote entity (Entity Y), the adaptive security appliance can enroll with the CA on behalf of Entity X (Cisco UP). In the enrollment request, the Entity X identity (domain name) is used.

Figure 47-3 shows the way to establish the trust relationship. The adaptive security appliance enrolls with the third party CA by using the Cisco UP FQDN as if the adaptive security appliance is the Cisco UP.

Figure 47-3 How the Security Appliance Represents Cisco Unified Presence – Certificate Impersonate



Security Certificate Exchange Between Cisco UP and the Security Appliance

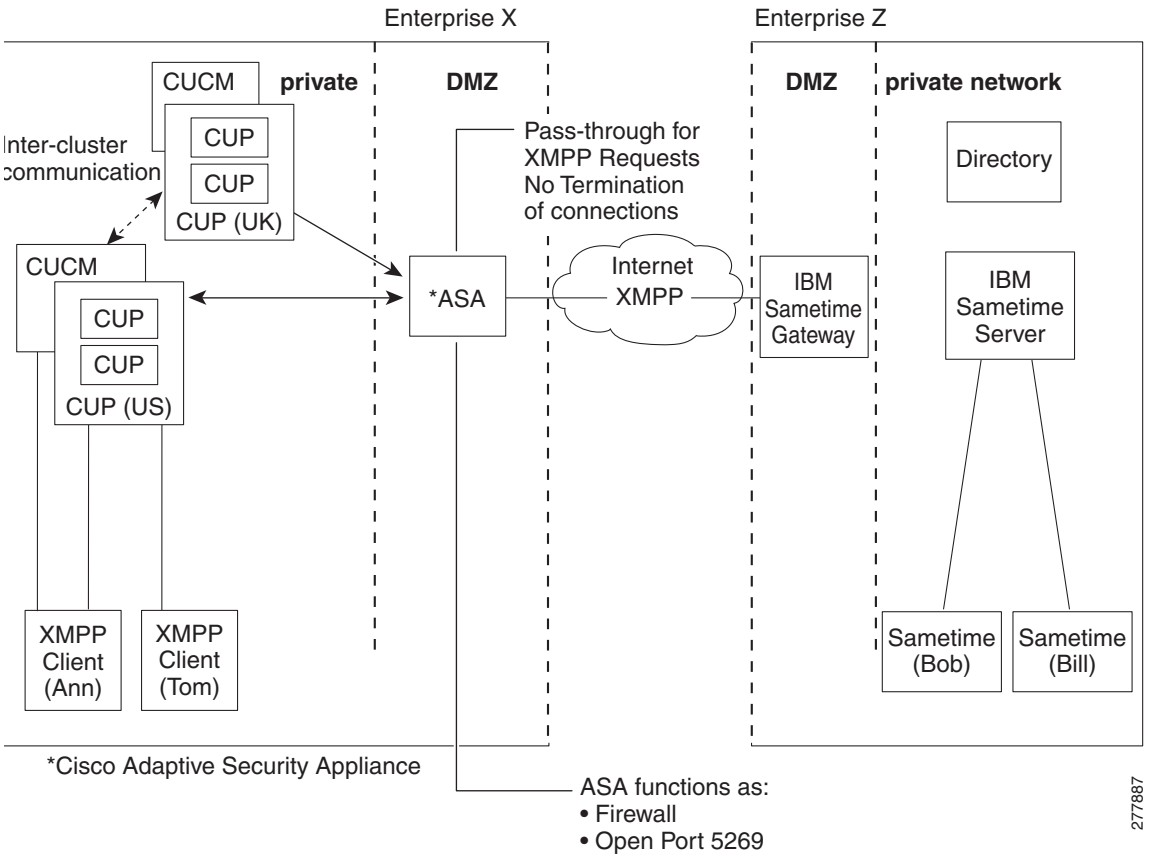
You need to generate the keypair for the certificate (such as `cup_proxy_key`) used by the adaptive security appliance, and configure a trustpoint to identify the self-signed certificate sent by the adaptive security appliance to Cisco UP (such as `cup_proxy`) in the TLS handshake.

For the adaptive security appliance to trust the Cisco UP certificate, you need to create a trustpoint to identify the certificate from the Cisco UP (such as `cert_from_cup`), and specify the enrollment type as terminal to indicate that you will paste the certificate received from the Cisco UP into the terminal.

XMPP Federation Deployments

Figure 47-4 provides an example of an XMPP federated network between Cisco Unified Presence enterprise deployment and an IBM Sametime enterprise deployment. TLS is optional for XMPP federation. adaptive security appliance acts only as a firewall for XMPP federation; it does not provide TLS proxy functionality or PAT for XMPP federation.

Figure 47-4 Basic XMPP Federated Network between Cisco Unified Presence and IBM Sametime



There are two DNS servers within the internal Cisco Unified Presence enterprise deployment. One DNS server hosts the Cisco Unified Presence private address. The other DNS server hosts the Cisco Unified Presence public address and a DNS SRV records for SIP federation (_sipfederationtle), and XMPP federation (_xmpp-server) with Cisco Unified Presence. The DNS server that hosts the Cisco Unified Presence public address is located in the local DMZ.

For further information about configuring Cisco Unified Presence Federation for XMPP Federation, see the *Integration Guide for Configuring Cisco Unified Presence Release 8.0 for Interdomain Federation*:

http://www.cisco.com/en/US/products/ps6837/products_installation_and_configuration_guides_list.html

Configuration Requirements for XMPP Federation

For XMPP Federation, adaptive security appliance acts as a firewall only. You must open port 5269 for both incoming and outgoing XMPP federated traffic on adaptive security appliance.

These are sample access lists to open port 5269 on adaptive security appliance.

Allow traffic from any address to any address on port 5269:

```
access-list ALLOW-ALL extended permit tcp any any eq 5269
```

Allow traffic from any address to any single node on port 5269:

```
access-list ALLOW-ALL extended permit tcp any host <private cup IP address> eq 5269
```

If you do not configure the access list above, and you publish additional XMPP federation nodes in DNS, you must configure access to each of these nodes, for example:

```
object network obj_host_<private cup ip address>
#host <private cup ip address>
object network obj_host_<private cup2 ip address>
#host <private cup2 ip address>
object network obj_host_<public cup ip address>
#host <public cup ip address>
....
```

Configure the following NAT commands:

```
nat (inside,outside) source static obj_host_<private cup1 IP> obj_host_<public cup IP>
service
obj_udp_source_eq_5269 obj_udp_source_eq_5269
nat (inside,outside) source static obj_host_<private cup1 IP> obj_host_<public cup IP>
service
obj_tcp_source_eq_5269 obj_tcp_source_eq_5269
```

If you publish a single public IP address in DNS, and use arbitrary ports, configure the following:

(This example is for two additional XMPP federation nodes)

```
nat (inside,outside) source static obj_host_<private cup2 ip> obj_host_<public cup IP>
service
obj_udp_source_eq_5269 obj_udp_source_eq_25269
nat (inside,outside) source static obj_host_<private cup2 ip> obj_host_<public cup IP>
service
obj_tcp_source_eq_5269 obj_tcp_source_eq_25269

nat (inside,outside) source static obj_host_<private cup3 ip> obj_host_<public cup IP>
service
obj_udp_source_eq_5269 obj_udp_source_eq_35269
nat (inside,outside) source static obj_host_<private cup3 ip> obj_host_<public cup IP>
service
obj_tcp_source_eq_5269 obj_tcp_source_eq_35269
```

If you publish multiple public IP addresses in DNS all using port 5269, configure the following:

(This example is for two additional XMPP federation nodes)

```
nat (inside,outside) source static obj_host_<private cup2 ip> obj_host_<public cup2 IP>
service
obj_udp_source_eq_5269 obj_udp_source_eq_5269
nat (inside,outside) source static obj_host_<private cup2 ip> obj_host_<public cup2 IP>
service
obj_tcp_source_eq_5269 obj_tcp_source_eq_5269

nat (inside,outside) source static obj_host_<private cup3 ip> obj_host_<public cup3 IP>
service
obj_udp_source_eq_5269 obj_udp_source_eq_5269
nat (inside,outside) source static obj_host_<private cup3 ip> obj_host_<public cup IP>
service
obj_tcp_source_eq_5269 obj_tcp_source_eq_5269
```

Licensing for Cisco Unified Presence

The Cisco Unified Presence feature supported by the adaptive security appliance require a Unified Communications Proxy license.

The following table shows the Unified Communications Proxy license details by platform:

Model	License Requirement
ASA 5505	Base License and Security Plus License: 2 sessions[1]. *Optional license: 24 sessions.*
ASA 5510	Base License and Security Plus License: 2 sessions[1]. *Optional licenses: 24, 50, or 100 sessions.*
ASA 5520	Base License: 2 sessions[1]. *Optional licenses: 24, 50, 100, 250, 500, 750, or 1000 sessions.*
ASA 5540	Base License: 2 sessions[1]. *Optional licenses: 24, 50, 100, 250, 500, 750, 1000, or 2000 sessions.*
ASA 5550	Base License: 2 sessions[1]. *Optional licenses: 24, 50, 100, 250, 500, 750, 1000, 2000, or 3000 sessions.*
ASA 5580	Base License: 2 sessions[1]. *Optional licenses: 24, 50, 100, 250, 500, 750, 1000, 2000, 3000, 5000, or 10,000 sessions.*[2]

1. The following applications use TLS proxy sessions for their connections. Each TLS proxy session used by these applications (and only these applications) is counted against the UC license limit:
   - Phone Proxy
   - Presence Federation Proxy
   - Encrypted Voice Inspection

   Other applications that use TLS proxy sessions do not count towards the UC limit, for example, Mobility Advantage Proxy (which does not require a license) and IME (which requires a separate IME license).

   Some UC applications might use multiple sessions for a connection. For example, if you configure a phone with a primary and backup Cisco Unified Communications Manager, there are 2 TLS proxy connections, so 2 UC Proxy sessions are used.

   You independently set the TLS proxy limit using the **tls-proxy maximum-sessions** command. To view the limits of your model, enter the **tls-proxy maximum-sessions ?** command. When you apply a UC license that is higher than the default TLS proxy limit, the adaptive security appliance automatically sets the TLS proxy limit to match the UC limit. The TLS proxy limit takes precedence over the UC license limit; if you set the TLS proxy limit to be less than the UC license, then you cannot use all of the sessions in your UC license.

   Note: For license part numbers ending in "K8" (for example, licenses under 250 users), TLS proxy sessions are limited to 1000. For license part numbers ending in "K9" (for example, licenses 250 users or larger), the TLS proxy limit depends on the configuration, up to the model limit. K8 and K9 refer to whether the license is restricted for export: K8 is unrestricted, and K9 is restricted.

   Note: If you clear the configuration (using the **clear configure all** command, for example), then the TLS proxy limit is set to the default for your model; if this default is lower than the UC license limit, then you see an error message to use the tls-proxy maximum-sessions command to raise the limit again . If you use failover and enter the **write standby** command on the primary unit to force a configuration synchronization, the **clear configure all** command is generated on the secondary unit automatically, so you may see the warning message on the secondary unit. Because the configuration synchronization restores the TLS proxy limit set on the primary unit, you can ignore the warning.

   You might also use SRTP encryption sessions for your connections:
   - For K8 licenses, SRTP sessions are limited to 250.
   - For K9 licenses, there is not limit.

   Note: Only calls that require encryption/decryption for media are counted towards the SRTP limit; if passthrough is set for the call, even if both legs are SRTP, they do not count towards the limit.
2. With the 10,000-session license, the total combined sessions can be 10,000, but the maximum number of Phone Proxy sessions is 5000.

For more information about licensing, see Chapter 3, "Managing Feature Licenses."

Configuring Cisco Unified Presence Proxy for SIP Federation

This section contains the following topics:

Task Flow for Configuring Cisco Unified Presence Federation Proxy for SIP Federation

To configure a Cisco Unified Presence/LCS Federation scenario with the adaptive security appliance as the TLS proxy where there is a single Cisco UP that is in the local domain and self-signed certificates are used between the Cisco UP and the adaptive security appliance (like the scenario shown in Figure 47-1), perform the following tasks.

Step 1 Create the following static NAT for the local domain containing the Cisco UP.

For the inbound connection to the local domain containing the Cisco UP, create static PAT by entering the following command:

```
hostname(config)# object network name
hostname(config-network-object)# host real_ip
hostname(config-network-object)# nat (real_ifc,mapped_ifc) static mapped_ip service {tcp | udp} real_port mapped_port
```

Note For each Cisco UP that could initiate a connection (by sending SIP SUBSCRIBE) to the foreign server, you must also configure static PAT by using a different set of PAT ports.

For outbound connections or the TLS handshake, use dynamic NAT or PAT. The adaptive security appliance SIP inspection engine takes care of the necessary translation (fixup).

```
hostname(config)# object network name
hostname(config-network-object)# subnet real_ip netmask
hostname(config-network-object)# nat (real_ifc,mapped_ifc) dynamic mapped_ip
```

For information about configuring NAT and PAT for the Cisco Presence Federation proxy, see Chapter 28, "Configuring Network Object NAT" and Chapter 29, "Configuring Twice NAT".

Step 2 Create the necessary RSA keypairs and proxy certificate, which is a self-signed certificate, for the remote entity. See Creating Trustpoints and Generating Certificates, page 47-9.

Step 3 Install the certificates. See Installing Certificates, page 47-10.

Step 4 Create the TLS proxy instance for the Cisco UP clients connecting to the Cisco UP server. See Creating the TLS Proxy Instance, page 47-12.

Step 5 Enable the TLS proxy for SIP inspection. See Enabling the TLS Proxy for SIP Inspection, page 47-13.

Creating Trustpoints and Generating Certificates

You need to generate the keypair for the certificate (such as `cup_proxy_key`) used by the adaptive security appliance, and configure a trustpoint to identify the self-signed certificate sent by the adaptive security appliance to Cisco UP (such as `cup_proxy`) in the TLS handshake.

	Command	Purpose
Step 1	hostname(config)# **crypto key generate rsa label** *key-pair-label* **modulus** *size* **Example:** crypto key generate rsa label ent_y_proxy_key modulus 1024 INFO: The name for the keys will be: ent_y_proxy_key Keypair generation process begin. Please wait... hostname(config)#	Creates the RSA keypair that can be used for the trustpoints. The keypair is used by the self-signed certificate presented to the local domain containing the Cisco UP (proxy for the remote entity).
Step 2	hostname(config)# **crypto ca trustpoint** *trustpoint_name* **Example:** hostname(config)# crypto ca trustpoint ent_y_proxy	Enters the trustpoint configuration mode for the specified trustpoint so that you can create the trustpoint for the remote entity. A trustpoint represents a CA identity and possibly a device identity, based on a certificate issued by the CA.
Step 3	hostname(config-ca-trustpoint)# **enrollment self**	Generates a self-signed certificate.
Step 4	hostname(config-ca-trustpoint)# **fqdn none**	Specifies not to include a fully qualified domain name (FQDN) in the Subject Alternative Name extension of the certificate during enrollment.
Step 5	hostname(config-ca-trustpoint)# **subject-name** *X.500_name* **Example:** hostname(config-ca-trustpoint)# subject-name cn=Ent-Y-Proxy	Includes the indicated subject DN in the certificate during enrollment
Step 6	hostname(config-ca-trustpoint)# **keypair** *keyname* **Example:** hostname(config-ca-trustpoint)# keypair ent_y_proxy_key	Specifies the key pair whose public key is to be certified.
Step 7	hostname(config-ca-trustpoint)# **exit**	Exits from the CA Trustpoint configuration mode.
Step 8	hostname(config)# **crypto ca enroll** *trustpoint* **Example:** hostname(config)# crypto ca enroll ent_y_proxy	Starts the enrollment process with the CA and specifies the name of the trustpoint to enroll with.

What to Do Next

Install the certificate on the local entity truststore. You could also enroll the certificate with a local CA trusted by the local entity. See the "Installing Certificates" section on page 47-10.

Installing Certificates

Export the self-signed certificate for the adaptive security appliance created in the "Creating Trustpoints and Generating Certificates" section on page 47-9 and install it as a trusted certificate on the local entity. This task is necessary for local entity to authenticate the adaptive security appliance.

Prerequisites

To create a proxy certificate on the adaptive security appliance that is trusted by the remote entity, obtain a certificate from a trusted CA. For information about obtaining a certificate from a trusted CA, see the "Configuring Digital Certificates" section on page 37-1.

	Command	Purpose
Step 1	hostname(config)# **crypto ca export** *trustpoint* **identity-certificate** **Example:** hostname(config)# crypto ca export ent_y_proxy identity-certificate	Export the adaptive security appliance self-signed (identity) certificate.
Step 2	hostname(config)# **crypto ca trustpoint** *trustpoint_name* **Example:** hostname(config)# crypto ca trustpoint ent_x_cert ! for Entity X's self-signed certificate	Enters the trustpoint configuration mode for the specified trustpoint so that you can create the trustpoint for the local entity. A trustpoint represents a CA identity and possibly a device identity, based on a certificate issued by the CA.
Step 3	hostname(config-ca-trustpoint)# **enrollment terminal**	Specifies cut and paste enrollment with this trustpoint (also known as manual enrollment). If the local entity uses a self-signed certificate, the self-signed certificate must be installed; if the local entity uses a CA-issued certificate, the CA certificate needs to be installed. This configuration shows the commands for using a self-signed certificate.
Step 4	hostname(config-ca-trustpoint)# **exit**	Exits from the CA Trustpoint configuration mode.
Step 5	hostname(config)# **crypto ca authenticate** *trustpoint* **Example:** hostname(config)# crypto ca authenticate ent_x_cert Enter the base 64 encoded CA certificate. End with a blank line or the word "quit" on a line by itself [certificate data omitted] Certificate has the following attributes: Fingerprint: 21B598D5 4A81F3E5 0B24D12E 3F89C2E4 % Do you accept this certificate? [yes/no]: yes Trustpoint CA certificate accepted. % Certificate successfully imported	Installs and authenticates the CA certificates associated with a trustpoint created for the local entity. Where *trustpoint* specifies the trustpoint from which to obtain the CA certificate. Maximum name length is 128 characters. The adaptive security appliance prompts you to paste the base-64 formatted CA certificate onto the terminal.
Step 6	hostname(config)# **crypto ca trustpoint** *trustpoint_name* **Example:** hostname(config)# crypto ca trustpoint ent_y_ca ! for Entity Y's CA certificate	Install the CA certificate that signs the remote entity certificate on the adaptive security appliance by entering the following commands. This step is necessary for the adaptive security appliance to authenticate the remote entity.
Step 7	hostname(config-ca-trustpoint)# **enrollment terminal**	Specifies cut and paste enrollment with this trustpoint (also known as manual enrollment).
Step 8	hostname(config-ca-trustpoint)# **exit**	Exits from the CA Trustpoint configuration mode.
Step 9	hostname(config)# **crypto ca authenticate** *trustpoint* **Example:** hostname(config)# crypto ca authenticate ent_y_ca Enter the base 64 encoded CA certificate. End with a blank line or the word "quit" on a line by itself MIIDRTCCAu+gAwIBAgIQKVcqP/KW74VP0NZzL+JbRTANBgkqhkiG9w0BAQUFADCB [certificate data omitted] /7QEM8izy0EOTSErKu7Nd76jwf5e4qttkQ==	Installs and authenticates the CA certificates associated with a trustpoint created for the local entity. The adaptive security appliance prompts you to paste the base-64 formatted CA certificate onto the terminal.

What to Do Next

Once you have created the trustpoints and installed the certificates for the local and remote entities on the adaptive security appliance, create the TLS proxy instance. See the "Creating the TLS Proxy Instance" section on page 47-12.

Creating the TLS Proxy Instance

Because either server can initiate the TLS handshake (unlike IP Telephony or Cisco Unified Mobility, where only the clients initiate the TLS handshake), you must configure by-directional TLS proxy rules. Each enterprise can have an adaptive security appliance as the TLS proxy.

Create TLS proxy instances for the local and remote entity initiated connections respectively. The entity that initiates the TLS connection is in the role of "TLS client". Because the TLS proxy has a strict definition of "client" and "server" proxy, two TLS proxy instances must be defined if either of the entities could initiate the connection.

	Command	Purpose
Step 1	! Local entity to remote entity hostname(config)# **tls-proxy** proxy_name **Example:** hostname(config)# tls-proxy ent_x_to_y	Creates the TLS proxy instance.
Step 2	hostname(config-tlsp)# **server trust-point** *proxy_name* **Example:** hostname(config-tlsp)# server trust-point ent_y_proxy	Specifies the proxy trustpoint certificate presented during TLS handshake. The certificate must be owned by the adaptive security appliance (identity certificate). Where the *proxy_name* for the **server trust-point** command is the remote entity proxy name.
Step 3	hostname(config-tlsp)# **client trust-point** *proxy_trustpoint* **Example:** hostname(config-tlsp)# client trust-point ent_x_proxy	Specifies the trustpoint and associated certificate that the adaptive security appliance uses in the TLS handshake when the adaptive security appliance assumes the role of the TLS client. The certificate must be owned by the adaptive security appliance (identity certificate). Where the *proxy_trustpoint* for the **client trust-point** command is the local entity proxy.
Step 4	hostname(config-tlsp)# **client cipher-suite** *cipher_suite* **Example:** hostname(config-tlsp)# client cipher-suite aes128-sha1 aes256-sha1 3des-sha1 null-sha1	Specifies cipher suite configuration. For client proxy (the proxy acts as a TLS client to the server), the user-defined cipher suite replaces the default cipher suite.
Step 5	! Remote entity to local entity hostname(config)# **tls-proxy** *proxy_name* **Example:** tls-proxy ent_y_to_x	Creates the TLS proxy instance.
Step 6	hostname(config-tlsp)# **server trust-point** *proxy_name* **Example:** hostname(config-tlsp)# server trust-point ent_x_proxy	Specifies the proxy trustpoint certificate presented during TLS handshake. Where the *proxy_name* for the **server trust-point** command is the local entity proxy name

	Command	Purpose

Step 7	`hostname(config-tlsp)# client trust-point proxy_trustpoint`
	Example:
	`hostname(config-tlsp)# client trust-point ent_y_proxy`
		Specifies the trustpoint and associated certificate that the adaptive security appliance uses in the TLS handshake when the adaptive security appliance assumes the role of the TLS client.
		Where the *proxy_trustpoint* for the **client trust-point** command is the remote entity proxy.

Step 8	`hostname(config-tlsp)# client cipher-suite cipher_suite`
	Example:
	`hostname(config-tlsp)# client cipher-suite aes128-sha1 aes256-sha1 3des-sha1 null-sha1`
		Specifies cipher suite configuration.

What to Do Next

Once you have created the TLS proxy instance, enable it for SIP inspection. See the "Enabling the TLS Proxy for SIP Inspection" section on page 47-13.

Enabling the TLS Proxy for SIP Inspection

Enable the TLS proxy for SIP inspection and define policies for both entities that could initiate the connection.

	Command	Purpose

Step 1	`hostname(config)# access-list id extended permit tcp host src_ip host dest_ip eq port`
	Examples:
	`access-list ent_x_to_y extended permit tcp host 10.0.0.2 host 192.0.2.254 eq 5061`
	`access-list ent_y_to_x extended permit tcp host 192.0.2.254 host 192.0.2.1 eq 5061`
		Adds an Access Control Entry. The access list is used to specify the class of traffic to inspect.

Step 2	`hostname(config)# class-map class_map_name`
	Example:
	`hostname(config)# class-map ent_x_to_y`
		Configures the secure SIP class of traffic to inspect.
		Where *class_map_name* is the name of the SIP class map.

Step 3	`hostname(config-cmap)# match access-list access_list_name`
	Example:
	`hostname(config-cmap)# match access-list ent_x_to_y`
		Identifies the traffic to inspect.

Step 4	`hostname(config-cmap)# exit`	Exits from Class Map configuration mode.

Step 5	`hostname(config)# policy-map type inspect sip policy_map_name`
	Example:
	`hostname(config)# policy-map type inspect sip sip_inspect`
		Defines special actions for SIP inspection application traffic.

Step 6	`hostname(config-pmap)# parameters` `! SIP inspection parameters`	Specifies the parameters for SIP inspection. Parameters affect the behavior of the inspection engine. The commands available in parameters configuration mode depend on the application.
Step 7	`hostname(config-pmap)# exit`	Exits from Policy Map configuration mode.

	Command	Purpose

Step 8	`hostname(config)# policy-map name`
	Example:
	`hostname(config)# policy-map global_policy`
		Configure the policy map and attach the action to the class of traffic.

Step 9	`hostname(config-pmap)# class classmap_name`
	Example:
	`hostname(config-pmap)# class ent_x_to_y`
		Assigns a class map to the policy map so that you can assign actions to the class map traffic.
		Where *classmap_name* is the name of the SIP class map.

Step 10	`hostname(config-pmap)# inspect sip sip_map tls-proxy proxy_name` `hostname(config-pmap)# inspect sip sip_inspect tls-proxy ent_x_to_y`	Enables TLS proxy for the specified SIP inspection session.
Step 11	`hostname(config-pmap)# exit`	Exits from Policy Map configuration mode.

Step 12	`hostname(config)# service-policy policy_map_name global`
	Example:
	`hostname(config)# service-policy global_policy global`
		Enables the service policy for SIP inspection for all interfaces.
		Where name for the policy-map command is the name of the global policy map.

Monitoring Cisco Unified Presence

Debugging is similar to debugging TLS proxy for IP Telephony. You can enable TLS proxy debug flags along with SSL syslogs to debug TLS proxy connection problems.

For example, use the following commands to enable TLS proxy-related debug and syslog output only:

```
hostname(config)# debug inspect tls-proxy events
hostname(config)# debug inspect tls-proxy errors
hostname(config)# logging enable
hostname(config)# logging timestamp
hostname(config)# logging list loglist message 711001
hostname(config)# logging list loglist message 725001-725014
hostname(config)# logging list loglist message 717001-717038
hostname(config)# logging buffer-size 1000000
hostname(config)# logging buffered loglist
hostname(config)# logging debug-trace
```

For information about TLS proxy debugging techniques and sample output, see the "Monitoring the TLS Proxy" section on page 45-14.

Enable the **debug sip** command for SIP inspection engine debugging. See the *Cisco ASA 5500 Series Command Reference*.

Additionally, you can capture the raw and decrypted data by the TLS proxy by entering the following commands:

```
hostname# capture mycap interface outside (capturing raw packets)
hostname# capture mycap-dec type tls-proxy interface outside (capturing decrypted data)
hostname# show capture capture_name
hostname# copy /pcap capture:capture_name tftp://tftp_location
```

Configuration Example for Cisco Unified Presence

This section contains the following topics:

Example Configuration for SIP Federation Deployments

The following sample illustrates the necessary configuration for the adaptive security appliance to perform TLS proxy for Cisco Unified Presence as shown in Figure 47-5. It is assumed that a single Cisco UP (Entity X) is in the local domain and self-signed certificates are used between Entity X and the ASA.

For each Cisco UP that could initiate a connection (by sending SIP SUBSCRIBE) to the foreign server, you must also configure static PAT and if you have another Cisco UP with the address (10.0.0.3 in this sample), it must use a different set of PAT ports (such as 45062 or 45070). Dynamic NAT or PAT can be used for outbound connections or TLS handshake. The adaptive security appliance SIP inspection engine takes care of the necessary translation (fixup).

When you create the necessary RSA key pairs, a key pair is used by the self-signed certificate presented to Entity X (proxy for Entity Y). When you create a proxy certificate for Entity Y, the certificate is installed on the Entity X truststore. It could also be enrolled with a local CA trusted by Entity X.

Exporting the adaptive security appliance self-signed certificate (ent_y_proxy) and installing it as a trusted certificate on Entity X is necessary for Entity X to authenticate the adaptive security appliance. Exporting the Entity X certificate and installing it on the adaptive security appliance is needed for the adaptive security appliance to authenticate Entity X during handshake with X. If Entity X uses a self-signed certificate, the self-signed certificate must be installed; if Entity X uses a CA issued the certificate, the CA's certificated needs to be installed.

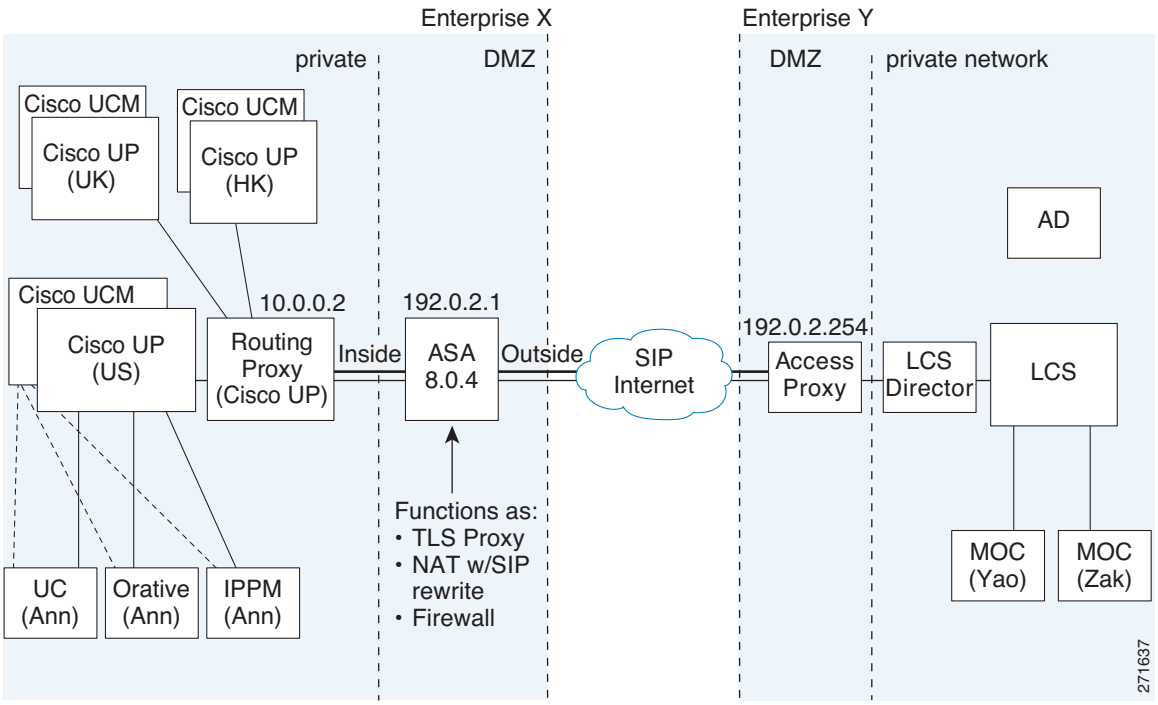
For information about obtaining a certificate from a trusted CA, see the "Configuring Digital Certificates" section on page 37-1.

Installing the CA certificate that signs the Entity Y certificate on the adaptive security appliance is necessary for the adaptive security appliance to authenticate Entity Y.

When creating TLS proxy instances for Entity X and Entity Y, the entity that initiates the TLS connection is in the role of "TLS client". Because the TLS proxy has strict definition of "client" and "server" proxy, two TLS proxy instances must be defined if either of the entities could initiate the connection.

When enabling the TLS proxy for SIP inspection, policies must be defined for both entities that could initiate the connection.

Figure 47-5 Typical Cisco Unified Presence/LCS Federation Scenario




```
object network obj-10.0.0.2-01
    host 10.0.0.2
    nat (inside,outside) static 192.0.2.1 service tcp 5061 5061
object network obj-10.0.0.2-02
    host 10.0.0.2
    nat (inside,outside) static 192.0.2.1 service tcp 5062 5062
object network obj-10.0.0.2-03
    host 10.0.0.2
    nat (inside,outside) static 192.0.2.1 service udp 5070 5070
object network obj-10.0.0.3-01
    host 10.0.0.3
    nat (inside,outside) static 192.0.2.1 service tcp 5062 45062
object network obj-10.0.0.3-02
    host 10.0.0.3
    nat (inside,outside) static 192.0.2.1 service udp 5070 45070
object network obj-0.0.0.0-01
    subnet 0.0.0.0 0.0.0.0
    nat (inside,outside) dynamic 192.0.2.1
crypto key generate rsa label ent_y_proxy_key modulus 1024
    ! for self-signed Entity Y proxy certificate
crypto ca trustpoint ent_y_proxy
    enrollment self
    fqdn none
    subject-name cn=Ent-Y-Proxy
    keypair ent_y_proxy_key
crypto ca enroll ent_y_proxy
crypto ca export ent_y_proxy identity-certificate
! for Entity X's self-signed certificate
crypto ca trustpoint ent_x_cert
    enrollment terminal
crypto ca authenticate ent_x_cert
Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself
    [ certificate data omitted ]
```

```
quit
! for Entity Y's CA certificate
crypto ca trustpoint ent_y_ca
    enrollment terminal
crypto ca authenticate ent_y_ca
Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself
MIIDRTCCAu+gAwIBAgIQKVcqP/KW74VP0NZzL+JbRTANBgkqhkiG9w0BAQUFADCB
    [ certificate data omitted ]
/7QEM8izy0EOTSErKu7Nd76jwf5e4qttkQ==
quit
! Entity X to Entity Y
tls-proxy ent_x_to_y
    server trust-point ent_y_proxy
    client trust-point ent_x_proxy
    client cipher-suite aes128-sha1 aes256-sha1 3des-sha1 null-sha1
! Entity Y to Entity X
tls-proxy ent_y_to_x
    server trust-point ent_x_proxy
    client trust-point ent_y_proxy
    client cipher-suite aes128-sha1 aes256-sha1 3des-sha1 null-sha1
access-list ent_x_to_y extended permit tcp host 10.0.0.2 host 192.0.2.254 eq 5061
access-list ent_y_to_x extended permit tcp host 192.0.2.254 host 192.0.2.1 eq 5061
class-map ent_x_to_y
    match access-list ent_x_to_y
class-map ent_y_to_x
    match access-list ent_y_to_x
policy-map type inspect sip sip_inspect
    parameters
        ! SIP inspection parameters
policy-map global_policy
    class ent_x_to_y
        inspect sip sip_inspect tls-proxy ent_x_to_y
    class ent_y_to_x
        inspect sip sip_inspect tls-proxy ent_y_to_x
service-policy global_policy global
```

Example Access List Configuration for XMPP Federation

Example 1: This example access list configuration allows from any address to any address on port 5269:

```
access-list ALLOW-ALL extended permit tcp any any eq 5269
```

Example 2: This example access list configuration allows from any address to any single XMPP federation node on port 5269. The following values are used in this example:

- Private XMPP federation Cisco Unified Presence Release 8.0 IP address = 1.1.1.1
- XMPP federation listening port = 5269

```
access-list ALLOW-ALL extended permit tcp any host 1.1.1.1 eq 5269
```

Example 3: This example access list configuration allows from any address to specific XMPP federation nodes published in DNS.

Note The public addresses are published in DNS, but the private addresses are configured in the access-list command.

The following values are used in this sample configuration:

• Private XMPP federation Cisco Unified Presence Release 8.0 IP address = 1.1.1.1

• Private second Cisco Unified Presence Release 8.0 IP address= 2.2.2.2

• Private third Cisco Unified Presence Release 7.x IP address = 3.3.3.3

• XMPP federation listening port = 5269

```
access-list ALLOW-ALL extended permit tcp any host 1.1.1.1 eq 5269
access-list ALLOW-ALL extended permit tcp any host 2.2.2.2 eq 5269
access-list ALLOW-ALL extended permit tcp any host 3.3.3.3 eq 5269
```

Example 4: This example access list configuration allows only from a specific federated domain interface to specific XMPP federation nodes published in DNS.

Note The public addresses are published in DNS, but the private addresses are configured in the access-list command.

The following values are used in this sample configuration:

- Private XMPP federation Cisco Unified Presence Release 8.0 IP address = 1.1.1.1
- Private second Cisco Unified Presence Release 8.0 IP address = 2.2.2.2
- Private third Cisco Unified Presence Release 7.x IP address = 3.3.3.3
- XMPP federation listening port = 5269
- External interface of the foreign XMPP enterprise = 100.100.100.100

```
access-list ALLOW-ALL extended permit tcp host 100.100.100.100 host 1.1.1.1 eq 5269
access-list ALLOW-ALL extended permit tcp host 100.100.100.100 host 2.2.2.2 eq 5269
access-list ALLOW-ALL extended permit tcp host 100.100.100.100 host 3.3.3.3 eq 5269
```

Example NAT Configuration for XMPP Federation

Example 1: Single node with XMPP federation enabled

The following values are used in this sample configuration:

- Public Cisco Unified Presence IP address = 10.10.10.10
- Private XMPP federation Cisco Unified Presence Release 8.0 IP address = 1.1.1.1
- XMPP federation listening port = 5269

```
nat (inside,outside) source static obj_host_1.1.1.1 obj_host_10.10.10.10 service
obj_udp_source_eq_5269 obj_udp_source_eq_5269
nat (inside,outside) source static obj_host_1.1.1.1 obj_host_10.10.10.10 service
obj_tcp_source_eq_5269 obj_tcp_source_eq_5269
```

Example 2: Multiple nodes with XMPP federation, each with a public IP address in DNS

The following values are used in this sample configuration:

- Public Cisco Unified Presence IP addresses = 10.10.10.10, 20.20.20.20, 30.30.30.30
- Private XMPP federation Cisco Unified Presence Release 8.0 IP address = 1.1.1.1
- Private second Cisco Unified Presence Release 8.0 IP address = 2.2.2.2

- Private third Cisco Unified Presence Release 7.x IP address = 3.3.3.3
- XMPP federation listening port = 5269

```
nat (inside,outside) source static obj_host_1.1.1.1 obj_host_10.10.10.10 service
obj_udp_source_eq_5269 obj_udp_source_eq_5269
nat (inside,outside) source static obj_host_1.1.1.1 obj_host_10.10.10.10 service
obj_tcp_source_eq_5269 obj_tcp_source_eq_5269

nat (inside,outside) source static obj_host_2.2.2.2 obj_host_20.20.20.20 service
obj_udp_source_eq_5269 obj_udp_source_eq_5269
nat (inside,outside) source static obj_host_2.2.2.2 obj_host_20.20.20.20 service
obj_tcp_source_eq_5269 obj_tcp_source_eq_5269

nat (inside,outside) source static obj_host_3.3.3.3 obj_host_30.30.30.30 service
obj_udp_source_eq_5269 obj_udp_source_eq_5269
nat (inside,outside) source static obj_host_3.3.3.3 obj_host_30.30.30.30 service
obj_tcp_source_eq_5269 obj_tcp_source_eq_5269
```

Example 3: Multiple nodes with XMPP federation, but a single public IP address in DNS with arbitrary ports published in DNS (PAT).

The following values are used in this sample configuration:

- Public Cisco Unified Presence IP Address = 10.10.10.10
- Private XMPP federation Cisco Unified Presence Release 8.0 IP address = 1.1.1.1, port 5269
- Private second Cisco Unified Presence Release 8.0 IP address = 2.2.2.2, arbitrary port 25269
- Private third Cisco Unified Presence Release 7.x IP address = 3.3.3.3, arbitrary port 35269

```
nat (inside,outside) source static obj_host_1.1.1.1 obj_host_10.10.10.10 service
obj_udp_source_eq_5269 obj_udp_source_eq_5269
nat (inside,outside) source static obj_host_1.1.1.1 obj_host_10.10.10.10 service
obj_tcp_source_eq_5269 obj_tcp_source_eq_5269

nat (inside,outside) source static obj_host_2.2.2.2 obj_host_10.10.10.10 service
obj_udp_source_eq_5269 obj_udp_source_eq_25269
nat (inside,outside) source static obj_host_2.2.2.2 obj_host_10.10.10.10 service
obj_tcp_source_eq_5269 obj_tcp_source_eq_25269

nat (inside,outside) source static obj_host_3.3.3.3 obj_host_10.10.10.10 service
obj_udp_source_eq_5269 obj_udp_source_eq_35269
nat (inside,outside) source static obj_host_3.3.3.3 obj_host_10.10.10.10 service
obj_tcp_source_eq_5269 obj_tcp_source_eq_35269
```

Feature History for Cisco Unified Presence

Table 47-1 lists the release history for this feature.

Table 47-1 Feature History for Cisco Phone Proxy

Feature Name	Releases	Feature Information
Cisco Presence Federation Proxy	8.0(4)	The Cisco Unified Presence proxy feature was introduced.
Cisco Presence Federation Proxy	8.3(1)	The Unified Communications Wizard was added to ASDM. By using the wizard, you can configure the Cisco Presence Federation Proxy. Support for XMPP Federation was introduced.