Cisco ASA Series Firewall CLI Configuration Guide

Software Version 9.4
For the ASA 5506-X, ASA 5506H-X, ASA 5506W-X, ASA 5508-X, ASA 5512-X, ASA 5515-X, ASA 5516-X, ASA 5525-X, ASA 5545-X, ASA 5555-X, ASA 5585-X, ASA Services Module, and the Adaptive Security Virtual Appliance

First Published: March 23, 2015
Last Updated: April 7, 2015

Cisco Systems, Inc.
www.cisco.com

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco website at www.cisco.com/go/offices.

Text Part Number: N/A, Online only

Cisco ASA Series Firewall CLI Configuration Guide

About This Guide

Document Objectives

The purpose of this guide is to help you configure the firewall features for Cisco ASA series using the command-line interface. This guide does not cover every feature, but describes only the most common configuration scenarios.

You can also configure and monitor the ASA by using the Adaptive Security Device Manager (ASDM), a web-based GUI application. ASDM includes configuration wizards to guide you through some common configuration scenarios, and online help for less common scenarios.

Throughout this guide, the term “ASA” applies generically to supported models, unless specified otherwise.

Related Documentation

For more information, see *Navigating the Cisco ASA Series Documentation* at http://www.cisco.com/go/asadocs.

Conventions

This document uses the following conventions:

Convention	Indication
bold font	Commands and keywords and user-entered text appear in **bold** font.
italic font	Document titles, new or emphasized terms, and arguments for which you supply values are in *italic* font.
[ ]	Elements in square brackets are optional.

{x | y | z }	Required alternative keywords are grouped in braces and separated by vertical bars.
[x | y | z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
`courier` font	Terminal sessions and information the system displays appear in `courier` font.
`courier bold` font	Commands and keywords and user-entered text appear in **`bold courier`** font.
`courier italic` font	Arguments for which you supply values are in *`courier italic`* font.
< >	Nonprinting characters such as passwords are in angle brackets.
[ ]	Default responses to system prompts are in square brackets.
!, #	An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.

Note Means *reader take note*.

Tip Means *the following information will help you solve a problem.*

Caution Means *reader be careful*. In this situation, you might perform an action that could result in equipment damage or loss of data.

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, using the Cisco Bug Search Tool (BST), submitting a service request, and gathering additional information, see *What's New in Cisco Product Documentation* at: http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html.

Subscribe to *What's New in Cisco Product Documentation*, which lists all new and revised Cisco technical documentation, as an RSS feed and deliver content directly to your desktop using a reader application. The RSS feeds are a free service.

Introduction to Cisco ASA Firewall Services

First Published: March 23, 2015
Last Updated: April 7, 2015

Firewall services are those ASA features that are focused on controlling access to the network, including services that block traffic and services that enable traffic flow between internal and external networks. These services include those that protect the network against threats, such as Denial of Service (DoS) and other attacks.

The following topics provide an overview of firewall services.

How to Implement Firewall Services

The following procedure provides a general sequence for implementing firewall services. However, each step is optional, needed only if you want to provide the service to your network.

Before You Begin

Configure the ASA according to the general operations configuration guide, including at minimum basic settings, interface configuration, routing, and management access.

Procedure

Step 1 Implement access control for the network. See Basic Access Control, page 1-2.

Step 2 Implement application filtering. See Application Filtering, page 1-2.

Step 3 Implement URL filtering. See URL Filtering, page 1-3.

Step 4 Implement threat protection. See Threat Protection, page 1-3.

Step 5 Implement Network Address Translation (NAT). See Network Address Translation, page 1-4.

Step 6 Implement application inspection if the default settings are insufficient for your network. See Application Inspection, page 1-5.

Basic Access Control

Access rules, applied per interface or globally, are your first line of defense. You can drop, upon entry, specific types of traffic, or traffic from (or to) specific hosts or networks. By default, the ASA allows traffic to flow freely from an inside network (higher security level) to an outside network (lower security level).

You can apply an access rule to limit traffic from inside to outside, or allow traffic from outside to inside.

Basic access rules control traffic using a "5-tuple" of source address and port, destination address and port, and protocol.

You can augment your rules by making them identity aware. This lets you configure rules based on user identity or group membership. To implement identity control, do any combination of the following:

- Install Cisco Context Directory Agent (CDA), also known as AD agent, on a separate server to collect user and group information already defined in your Active Directory (AD) server. Then, configure the ASA to get this information, and add user or group criteria to your access rules.
- Install Cisco Identity Services Engine (ISE) on a separate server to implement Cisco Trustsec. You can then add security group criteria to your access rules.
- Install the ASA FirePOWER module on the ASA and implement identity policies in the module. The identity-aware access policies in ASA FirePOWER would apply to any traffic that you redirect to the module.

Related Topics

- Access Control Lists, page 3-1
- Access Rules, page 4-1
- Identity Firewall, page 5-1
- ASA and Cisco TrustSec, page 6-1
- ASA FirePOWER Module, page 7-1

Application Filtering

The wide-spread use of web-based applications means that a lot of traffic runs over the HTTP or HTTPS protocols. With traditional 5-tuple access rules, you either allow or disallow all HTTP/HTTPS traffic. You might require more granular control of web traffic.

You can install a module on the ASA to provide application filtering to selectively allow HTTP or other traffic based on the application being used. Thus, you do not have to make a blanket permit for HTTP. You can look inside the traffic and prevent applications that are unacceptable for your network (for example, inappropriate file sharing). When you add a module for application filtering, do not configure HTTP inspection on the ASA.

To implement application filtering, install the ASA FirePOWER module on the ASA and use application filtering criteria in your ASA FirePOWER access rules. These policies apply to any traffic that you redirect to the module.

Related Topics

- ASA FirePOWER Module, page 7-1

URL Filtering

URL filtering denies or allows traffic based on the URL of the destination site.

The purpose of URL filtering is primarily to completely block or allow access to a web site. Although you can target individual pages, you typically specify a host name (such as www.example.com) or a URL category, which defines a list of host names that provide a particular type of service (such as Gambling).

When trying to decide whether to use URL filtering or application filtering for HTTP/HTTPS traffic, consider whether your intention is to create a policy that applies to all traffic directed at a web site. If your intention is to treat all such traffic the same way (denying it or allowing it), use URL filtering. If your intention is to selectively block or allow traffic to the site, use application filtering.

To implement URL filtering, do one of the following:

- Subscribe to the Cloud Web Security service, where you configure your filtering policies in ScanCenter, and then configure the ASA to send traffic to your Cloud Web Security account.
- Install the ASA FirePOWER module on the ASA and use URL filtering criteria in your ASA FirePOWER access rules. These policies apply to any traffic that you redirect to the module.

Related Topics

- ASA and Cisco Cloud Web Security, page 8-1
- ASA FirePOWER Module, page 7-1

Threat Protection

You can implement a number of measures to protect against scanning, denial of service (DoS), and other attacks. A number of ASA features help protect against attacks by applying connection limits and dropping abnormal TCP packets. Some features are automatic, others are configurable but have defaults appropriate in most cases, while others are completely optional and you must configure them if you want them.

Following are the threat protection services available with the ASA.

- IP packet fragmentation protection—The ASA performs full reassembly of all ICMP error messages and virtual reassembly of the remaining IP fragments that are routed through the ASA, and drops fragments that fail the security check. No configuration is necessary.
- Connection limits, TCP normalization, and other connection-related features—Configure connection-related services such as TCP and UDP connection limits and timeouts, TCP sequence number randomization, TCP normalization, and TCP state bypass. TCP normalization is designed to drop packets that do not appear normal.

For example, you can limit TCP and UDP connections and embryonic connections (a connection request that has not finished the necessary handshake between source and destination). Limiting the number of connections and embryonic connections protects you from a DoS attack. The ASA uses the embryonic limit to trigger TCP Intercept, which protects inside systems from a DoS attack perpetrated by flooding an interface with TCP SYN packets.

- Threat detection—Implement threat detection on the ASA to collect statistics to help identify attacks. Basic threat detection is enabled by default, but you can implement advanced statistics and scanning threat detection. You can shun hosts that are identified as a scanning threat.
- Next-Generation IPS—Install the ASA FirePOWER module on the ASA and implement Next Generation IPS intrusion rules in your ASA FirePOWER. These policies would apply to any traffic that you redirect to ASA FirePOWER.

Related Topics

- Connection Settings, page 16-1
- Threat Detection, page 18-1
- ASA FirePOWER Module, page 7-1

Network Address Translation

One of the main functions of Network Address Translation (NAT) is to enable private IP networks to connect to the Internet. NAT replaces a private IP address with a public IP address, translating the private addresses in the internal private network into legal, routable addresses that can be used on the public Internet. In this way, NAT conserves public addresses because you can advertise at a minimum only one public address for the entire network to the outside world.

Other functions of NAT include:

- Security—Keeping internal IP addresses hidden discourages direct attacks.
- IP routing solutions—Overlapping IP addresses are not a problem when you use NAT.
- Flexibility—You can change internal IP addressing schemes without affecting the public addresses available externally; for example, for a server accessible to the Internet, you can maintain a fixed IP address for Internet use, but internally, you can change the server address.
- Translating between IPv4 and IPv6 (Routed mode only)—If you want to connect an IPv6 network to an IPv4 network, NAT lets you translate between the two types of addresses.

NAT is not required. If you do not configure NAT for a given set of traffic, that traffic will not be translated, but will have all of the security policies applied as normal.

Related Topics

- Network Address Translation (NAT), page 9-1
- NAT Examples and Reference, page 10-1

Application Inspection

Application inspection engines are required for services that embed IP addressing information in the user data packet or that open secondary channels on dynamically assigned ports. These protocols require the ASA to do a deep packet inspection, to open the required pinholes and to apply network address translation (NAT).

The default ASA policy already applies inspection globally for many popular protocols, such as DNS, FTP, SIP, ESMTP, TFTP, and others. The default inspections might be all you require for your network.

However, you might need to enable inspection for other protocols, or fine-tune an inspection. Many inspections include detailed options that let you control packets based on their contents. If you know a protocol well, you can apply fine-grained control on that traffic.

You use service policies to configure application inspection. You can configure a global service policy, or apply a service policy to each interface, or both.

Related Topics

- Service Policy Using the Modular Policy Framework, page 11-1
- Getting Started with Application Layer Protocol Inspection, page 12-1
- Inspection of Basic Internet Protocols, page 13-1
- Inspection for Voice and Video Protocols, page 14-1
- Inspection of Database, Directory, and Management Protocols, page 15-1

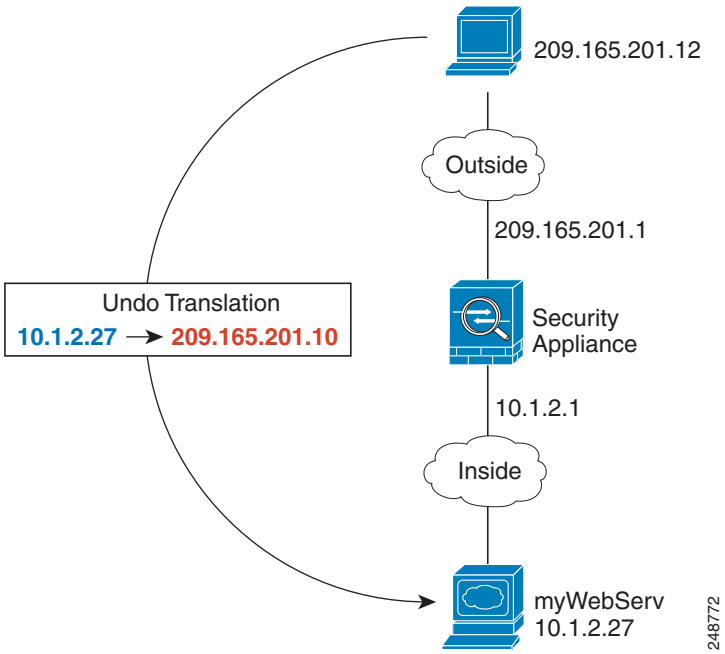
Use Case: Expose a Server to the Public

You can make certain application services on a server available to the public. For example, you could expose a web server, so that users can connect to the web pages but not make any other connections to the server.

To expose a server to the public, you typically need to create access rules that allow the connection and NAT rules to translate between the server's internal IP address and an external address that the public can use. In addition, you can use port address translation (PAT) to map an internal port to an external port, if you do not want the externally exposed service to use the same port as the internal server. For example, if the internal web server is not running on TCP/80, you can map it to TCP/80 to make connections easier for external users.

The following example makes a web server on the inside private network available for public access.

Figure 1-1 Static NAT for an Inside Web Server



Procedure

Step 1 Create a network object for the internal web server.

```
hostname(config)# object network myWebServ
hostname(config-network-object)# host 10.1.2.27
```

Step 2 Configure static NAT for the object:

```
hostname(config-network-object)# nat (inside,outside) static 209.165.201.10
```

Step 3 Add an access rule to the access group attached to the outside interface to permit web access to the server.

```
hostname(config)# access-list outside_access_in line 1 extended permit tcp any4 object
myWebServ eq http
```

Step 4 If you do not already have an access group on the outside interface, apply it using the access-group command:

```
hostname(config)# access-group outside_access_in in interface outside
```

Related Topics

- Static NAT, page 9-27

PART 1

Access Control

Objects for Access Control

Objects are reusable components for use in your configuration. You can define and use them in Cisco ASA configurations in the place of inline IP addresses, services, names, and so on. Objects make it easy to maintain your configurations because you can modify an object in one place and have it be reflected in all other places that are referencing it. Without objects you would have to modify the parameters for every feature when required, instead of just once. For example, if a network object defines an IP address and subnet mask, and you want to change the address, you only need to change it in the object definition, not in every feature that refers to that IP address.

Guidelines for Objects

IPv6 Guidelines

Supports IPv6 with the following restrictions:

- The ASA does not support IPv6 nested network object groups, so you cannot group an object with IPv6 entries under another IPv6 object group.
- You can mix IPv4 and IPv6 entries in a network object group; you cannot use a mixed object group for NAT.

Additional Guidelines and Limitations

- Objects must have unique names, because objects and object groups share the same name space. While you might want to create a network object group named “Engineering” and a service object group named “Engineering,” you need to add an identifier (or “tag”) to the end of at least one object group name to make it unique. For example, you can use the names “Engineering_admins” and “Engineering_hosts” to make the object group names unique and to aid in identification.
- Object names are limited to 64 characters, including letters, numbers, and these characters: .!@#$%^&()-_{}. Object names are case-sensitive.
- You cannot remove an object or make an object empty if it is used in a command, unless you enable forward referencing (the **forward-reference enable** command).

Configure Objects

The following sections describe how to configure objects that are primarily used on access control.

Configure Network Objects and Groups

Network objects and groups identify IP addresses or host names. Use these objects in access control lists to simplify your rules.

Configure a Network Object

A network object can contain a host, a network IP address, a range of IP addresses, or a fully qualified domain name (FQDN).

You can also enable NAT rules on the object (excepting FQDN objects). See the firewall configuration guide for more information about configuring object NAT.

Procedure

Step 1 Create or edit a network object using the object name.

```
hostname(config)# object network object_name
```

Example

```
hostname(config)# object network email-server
```

Step 2 Add an address to the object using one of the following commands. Use the **no** form of the command to remove the object.

- **host** {*IPv4_address* | *IPv6_address*}—The IPv4 or IPv6 address of a single host. For example, 10.1.1.1 or 2001:DB8::0DB8:800:200C:417A.
- **subnet** {*IPv4_address IPv4_mask* | *IPv6_address*/*IPv6_prefix*}—The address of a network. For IPv4 subnets, include the mask after a space, for example, 10.0.0.0 255.0.0.0. For IPv6, include the address and prefix as a single unit (no spaces), such as 2001:DB8:0:CD30::/60.
- **range** *start_address end_address*—A range of addresses. You can specify IPv4 or IPv6 ranges. Do not include masks or prefixes.
- **fqdn** [**v4** | **v6**] *fully_qualified_domain_name*—A fully-qualified domain name, that is, the name of a host, such as www.example.com. Specify **v4** to limit the address to IPv4, and **v6** for IPv6. If you do not specify an address type, IPv4 is assumed.

Example

```
hostname(config-network-object)# host 10.2.2.2
```

Step 3 (Optional) Add a description.

```
hostname(config-network-object)# description string
```

Configure a Network Object Group

Network object groups can contain multiple network objects as well as inline networks or hosts. Network object groups can include a mix of both IPv4 and IPv6 addresses.

However, you cannot use a mixed IPv4 and IPv6 object group for NAT, or object groups that include FQDN objects.

Procedure

Step 1 Create or edit a network object group using the object name.

```
ciscoasa(config)# object-group network group_name
```

Example

```
hostname(config)# object-group network admin
```

Step 2 Add objects and addresses to the network object group using one or more of the following commands. Use the **no** form of the command to remove an object.

- **network-object host** {*IPv4_address* | *IPv6_address*}—The IPv4 or IPv6 address of a single host. For example, 10.1.1.1 or 2001:DB8::0DB8:800:200C:417A.
- **network-object** {*IPv4_address IPv4_mask* | *IPv6_address*/*IPv6_prefix*}—The address of a network or host. For IPv4 subnets, include the mask after a space, for example, 10.0.0.0 255.0.0.0. For IPv6, include the address and prefix as a single unit (no spaces), such as 2001:DB8:0:CD30::/60.
- **network-object object** *object_name*—The name of an existing network object.
- **group-object** *object_group_name*—The name of an existing network object group.

Example

```
hostname(config-network-object-group)# network-object 10.1.1.0 255.255.255.0
hostname(config-network-object-group)# network-object 2001:db8:0:cd30::/60
hostname(config-network-object-group)# network-object host 10.1.1.1
hostname(config-network-object-group)# network-object host 2001:DB8::0DB8:800:200C:417A
hostname(config-network-object-group)# network-object object existing-object-1
hostname(config-network-object-group)# group-object existing-network-object-group
```

Step 3 (Optional) Add a description.

```
hostname(config-network-object-group)# description string
```

Example

To create a network group that includes the IP addresses of three administrators, enter the following commands:

```
hostname (config)# object-group network admins
hostname (config-protocol)# description Administrator Addresses
```

```
hostname (config-protocol)# network-object host 10.2.2.4
hostname (config-protocol)# network-object host 10.2.2.78
hostname (config-protocol)# network-object host 10.2.2.34
```

Create network object groups for privileged users from various departments by entering the following commands:

```
hostname (config)# object-group network eng
hostname (config-network)# network-object host 10.1.1.5
hostname (config-network)# network-object host 10.1.1.9
hostname (config-network)# network-object host 10.1.1.89

hostname (config)# object-group network hr
hostname (config-network)# network-object host 10.1.2.8
hostname (config-network)# network-object host 10.1.2.12

hostname (config)# object-group network finance
hostname (config-network)# network-object host 10.1.4.89
hostname (config-network)# network-object host 10.1.4.100
```

You then nest all three groups together as follows:

```
hostname (config)# object-group network admin
hostname (config-network)# group-object eng
hostname (config-network)# group-object hr
hostname (config-network)# group-object finance
```

Configure Service Objects and Service Groups

Service objects and groups identify protocols and ports. Use these objects in access control lists to simplify your rules.

- Configure a Service Object, page 2-4
- Configure a Service Group, page 2-5

Configure a Service Object

A service object can contain a single protocol, ICMP, ICMPv6, TCP or UDP port or port ranges.

Procedure

Step 1 Create or edit a service object using the object name.

```
ciscoasa(config)# object service object_name
```

Example

```
hostname(config)# object service web
```

Step 2 Add a service to the object using one of the following commands. Use the **no** form of the command to remove an object.

- **service** *protocol*—The name or number (0-255) of an IP protocol. Specify **ip** to apply to all protocols.

- **service** {**icmp** | **icmp6**} [*icmp-type* [*icmp_code*]]—For ICMP or ICMP version 6 messages. You can optionally specify the ICMP type by name or number (0-255) to limit the object to that message type. If you specify a type, you can optionally specify an ICMP code for that type (1-255). If you do not specify the code, then all codes are used.
- **service** {**tcp** | **udp**} [**source** *operator port*] [**destination** *operator port*]—For TCP or UDP. You can optionally specify ports for the source, destination, or both. You can specify the port by name or number. The operator can be one of the following:
  - **lt**—less than.
  - **gt**—greater than.
  - **eq**—equal to.
  - **neq**—not equal to.
  - **range**—an inclusive range of values. When you use this operator, specify two port numbers, for example, **range 100 200**.

Example

```
hostname(config-service-object)# service tcp destination eq http
```

Step 3 (Optional) Add a description.

```
hostname(config-service-object)# description string
```

Configure a Service Group

A service object group includes a mix of protocols, if desired, including optional source and destination ports for TCP or UDP.

Before You Begin

You can model all services using the generic service object group, which is explained here. However, you can still configure the types of service group objects that were available prior to ASA 8.3(1). These legacy objects include TCP/UDP/TCP-UDP port groups, protocol groups, and ICMP groups. The contents of these groups are equivalent to the associated configuration in the generic service object group, with the exception of ICMP groups, which do not support ICMP6 or ICMP codes. If you still want to use these legacy objects, for detailed instructions, see the **object-service** command description in the command reference on Cisco.com.

Procedure

Step 1 Create or edit a service object group using the object name.

```
ciscoasa(config)# object-group service group_name
```

Example

```
hostname(config)# object-group service general-services
```

Step 2 Add objects and services to the service object group using one or more of the following commands. Use the **no** form of the command to remove an object.

- **service-object** *protocol*—The name or number (0-255) of an IP protocol. Specify **ip** to apply to all protocols.

- **service-object** {**icmp** | **icmp6**} [*icmp-type* [*icmp_code*]]—For ICMP or ICMP version 6 messages. You can optionally specify the ICMP type by name or number (0-255) to limit the object to that message type. If you specify a type, you can optionally specify an ICMP code for that type (1-255). If you do not specify the code, then all codes are used.
- **service-object** {**tcp** | **udp** | **tcp-udp**} [**source** *operator port*] [**destination** *operator port*]—For TCP, UDP, or both. You can optionally specify ports for the source, destination, or both. You can specify the port by name or number. The operator can be one of the following:
  - **lt**—less than.
  - **gt**—greater than.
  - **eq**—equal to.
  - **neq**—not equal to.
  - **range**—an inclusive range of values. When you use this operator, specify two port numbers, for example, **range 100 200**.
- **service-object object** *object_name*—The name of an existing service object.
- **group-object** *object_group_name*—The name of an existing service object group.

Example

```
hostname(config-service-object-group)# service-object ipsec
hostname(config-service-object-group)# service-object tcp destination eq domain
hostname(config-service-object-group)# service-object icmp echo
hostname(config-service-object-group)# service-object object my-service
hostname(config-service-object-group)# group-object Engineering_groups
```

Step 3 (Optional) Add a description.

```
hostname(config-service-object-group)# description string
```

Examples

The following example shows how to add both TCP and UDP services to a service object group:

```
hostname(config)# object-group service CommonApps
hostname(config-service-object-group)# service-object tcp destination eq ftp
hostname(config-service-object-group)# service-object tcp-udp destination eq www
hostname(config-service-object-group)# service-object tcp destination eq h323
hostname(config-service-object-group)# service-object tcp destination eq https
hostname(config-service-object-group)# service-object udp destination eq ntp
```

The following example shows how to add multiple service objects to a service object group:

```
hostname(config)# object service SSH
hostname(config-service-object)# service tcp destination eq ssh
hostname(config)# object service EIGRP
hostname(config-service-object)# service eigrp
hostname(config)# object service HTTPS
hostname(config-service-object)# service tcp source range 1 1024 destination eq https
hostname(config)# object-group service Group1
hostname(config-service-object-group)# service-object object SSH
hostname(config-service-object-group)# service-object object EIGRP
hostname(config-service-object-group)# service-object object HTTPS
```

Configure Local User Groups

You can create local user groups for use in features that support the identity firewall by including the group in an extended ACL, which in turn can be used in an access rule, for example.

The ASA sends an LDAP query to the Active Directory server for user groups globally defined in the Active Directory domain controller. The ASA imports these groups for identity-based rules. However, the ASA might have localized network resources that are not defined globally that require local user groups with localized security policies. Local user groups can contain nested groups and user groups that are imported from Active Directory. The ASA consolidates local and Active Directory groups.

A user can belong to local user groups and user groups imported from Active Directory.

Because you can use usernames and user group names directly in an ACL, you need to configure local user groups only if:

- You want to create a group of users defined in the LOCAL database.
- You want to create a group of users or user groups that are not captured in a single user group defined on the AD server.

For information on how to enable the identity firewall, see Chapter 5, "Identity Firewall."

Procedure

Step 1 Create or edit a user object group using the object name.

```
hostname(config)# object-group user group_name
```

Example

```
hostname(config)# object-group user admins
```

Step 2 Add users and groups to the user object group using one or more of the following commands. Use the **no** form of the command to remove an object.

- **user** [*domain_NETBIOS_name*\]*username*—A username. If there is a space in the domain name or username, you must enclose the domain name and user name in quotation marks. The domain name can be LOCAL (for users defined in the local database) or an Active Directory (AD) domain name as specified in the **user-identity domain** *domain_NetBIOS_name* **aaa-server** *aaa_server_group_tag* command. When adding users defined in an AD domain, the *user_name* must be the Active Directory sAMAccountName, which is unique, instead of the common name (cn), which might not be unique. If you do not specify a domain name, the default is used, which is either LOCAL or the one defined on the **user-identity default-domain** command.
- **user-group** [*domain_NETBIOS_name*\\]*username*—A user group. If there is a space in the domain name or group name, you must enclose the domain name and group name in quotation marks. Note the double \\ that separates the domain and group names.
- **group-object** *object_group_name*—The name of an existing user object group.

Example

```
hostname(config-user-object-group)# user EXAMPLE\admin
hostname(config-user-object-group)# user-group EXAMPLE\\managers
hostname(config-user-object-group)# group-object local-admins
```

Step 3 (Optional) Add a description.

```
hostname(config-user-object-group)# description string
```

Configure Security Group Object Groups

You can create security group object groups for use in features that support Cisco TrustSec by including the group in an extended ACL, which in turn can be used in an access rule, for example.

When integrated with Cisco TrustSec, the ASA downloads security group information from the ISE. The ISE acts as an identity repository, by providing Cisco TrustSec tag-to-user identity mapping and Cisco TrustSec tag-to-server resource mapping. You provision and manage security group ACLs centrally on the ISE.

However, the ASA might have localized network resources that are not defined globally that require local security groups with localized security policies. Local security groups can contain nested security groups that are downloaded from the ISE. The ASA consolidates local and central security groups.

To create local security groups on the ASA, you create a local security object group. A local security object group can contain one or more nested security object groups or Security IDs or security group names. You can also create a new Security ID or security group name that does not exist on the ASA.

You can use the security object groups you create on the ASA to control access to network resources. You can use the security object group as part of an access group or service policy.

For information on how to integrate the ASA with Trustsec, see Chapter 6, "ASA and Cisco TrustSec."

Tip If you create a group with tags or names that are not known to the ASA, any rules that use the group will be inactive until the tags or names are resolved with ISE.

Procedure

Step 1 Create or edit a security group object group using the object name.

```
hostname(config)# object-group security group_name
```

Example

```
hostname(config)# object-group security mktg-sg
```

Step 2 Add objects to the service group object group using one or more of the following commands. Use the **no** form of the command to remove an object.

- **security-group** {**tag** *sgt_number* | **name** *sg_name*}—A security group tag (SGT) or name. A tag is a number from 1 to 65533 and is assigned to a device through IEEE 802.1X authentication, web authentication, or MAC authentication bypass (MAB) by the ISE. Security group names are created on the ISE and provide user-friendly names for security groups. The security group table maps SGTs to security group names. Consult your ISE configuration for the valid tags and names.
- **group-object** *object_group_name*—The name of an existing security group object group.

Example

```
hostname(config-security-object-group)# security-group tag 1
hostname(config-security-object-group)# security-group name mgkt
hostname(config-security-object-group)# group-object local-sg
```

Step 3 (Optional) Add a description.

```
hostname(config-security-object-group)# description string
```

Configure Time Ranges

A time range object defines a specific time consisting of a start time, an end time, and optional recurring entries. You use these objects on ACL rules to provide time-based access to certain features or assets. For example, you could create an access rule that allows access to a particular server during working hours only.

Note You can include multiple periodic entries in a time range object. If a time range has both absolute and periodic values specified, then the periodic values are evaluated only after the absolute start time is reached, and they are not further evaluated after the absolute end time is reached.

Creating a time range does not restrict access to the device. This procedure defines the time range only. You must then use the object in an access control rule.

Procedure

Step 1 Create the time range.

```
time-range name
```

Step 2 (Optional.) Add a start or end time (or both) to the time range.

```
absolute [start time date] [end time date]
```

If you do not specify a start time, the default start time is now.

The *time* is in the 24-hour format *hh*:*mm*. For example, 8:00 is 8:00 a.m. and 20:00 is 8:00 p.m.

The *date* is in the format *day month year*; for example, **1 January 2014**.

Step 3 (Optional.) Add recurring time periods.

```
periodic days-of-the-week time to [days-of-the-week] time
```

You can specify the following values for *days-of-the-week*. Note that you can specify a second day of the week only if you specify a single day for the first argument.

- **Monday**, **Tuesday**, **Wednesday**, **Thursday**, **Friday**, **Saturday**, or **Sunday**. You can specify more than one of these, separated by spaces, for the first *days-of-the-week* argument.
- **daily**
- **weekdays**
- **weekend**

The *time* is in the 24-hour format *hh*:*mm*. For example, 8:00 is 8:00 a.m. and 20:00 is 8:00 p.m.

You can repeat this command to configure more than one recurring period.

Examples

The following is an example of an absolute time range beginning at 8:00 a.m. on January 1, 2006. Because no end time and date are specified, the time range is in effect indefinitely.

```
hostname(config)# time-range for2006
hostname(config-time-range)# absolute start 8:00 1 january 2006
```

The following is an example of a weekly periodic time range from 8:00 a.m. to 6:00 p.m on weekdays:

```
hostname(config)# time-range workinghours
hostname(config-time-range)# periodic weekdays 8:00 to 18:00
```

The following example establishes an end date for the time range, and sets a weekday period from 8 a.m. to 5 p.m., plus different hours after 5 for Monday, Wednesday, Friday compared to Tuesday, Thursday.

```
asa4(config)# time-range contract-A-access
asa4(config-time-range)# absolute end 12:00 1 September 2025
asa4(config-time-range)# periodic weekdays 08:00 to 17:00
asa4(config-time-range)# periodic Monday Wednesday Friday 18:00 to 20:00
asa4(config-time-range)# periodic Tuesday Thursday 17:30 to 18:30
```

Monitoring Objects

To monitor objects and groups, enter the following commands:

- **show access-list**

  Displays the access list entries. Entries that include objects are also expanded out into individual entries based on the object contents.

- **show running-config object** [**id** *object_id*]

  Displays all current objects. Use the **id** keyword to view a single object by name.

- **show running-config object** *object_type*

  Displays the current objects by their type, **network** or **service**.

- **show running-config object-group** [**id** *group_id*]

  Displays all current object groups. Use the **id** keyword to view a single object group by name.

- **show running-config object-group** *grp_type*

  Displays the current object groups by their group type.

History for Objects

Feature Name	Platform Releases	Description
Object groups	7.0(1)	Object groups simplify ACL creation and maintenance. We introduced or modified the following commands: **object-group** *protocol*, **object-group** *network*, **object-group** *service*, **object-group** *icmp_type*.
Regular expressions and policy maps	7.2(1)	Regular expressions and policy maps were introduced to be used under inspection policy maps. The following commands were introduced: **class-map type regex**, **regex**, **match regex**.
Objects	8.3(1)	Object support was introduced. We introduced or modified the following commands: **object-network**, **object-service**, **object-group** *network*, **object-group** *service*, **network object**, **access-list extended**, **access-list webtype**, **access-list remark**.
User Object Groups for Identity Firewall	8.4(2)	User object groups for identity firewall were introduced. We introduced the following commands: **object-network user**, **user**.
Security Group Object Groups for Cisco TrustSec	8.4(2)	Security group object groups for Cisco TrustSec were introduced. We introduced the following commands: **object-network security**, **security**.
Mixed IPv4 and IPv6 network object groups	9.0(1)	Previously, network object groups could only contain all IPv4 addresses or all IPv6 addresses. Now network object groups can support a mix of both IPv4 and IPv6 addresses. **Note** You cannot use a mixed object group for NAT. We modified the following commands: **object-group network**.
Extended ACL and object enhancement to filter ICMP traffic by ICMP code	9.0(1)	ICMP traffic can now be permitted/denied based on ICMP code. We introduced or modified the following commands: **access-list extended**, **service-object**, **service**.

Access Control Lists

Access control lists (ACLs) are used by many different features. When applied to interfaces or globally as access rules, they permit or deny traffic that flows through the appliance. For other features, the ACL selects the traffic to which the feature will apply, performing a matching service rather than a control service.

The following sections explain the basics of ACLs and how to configure and monitor them. Access rules, ACLs applied globally or to interfaces, are explained in more detail in the firewall configuration guide.

About ACLs

Access control lists (ACLs) identify traffic flows by one or more characteristics, including source and destination IP address, IP protocol, ports, EtherType, and other parameters, depending on the type of ACL. ACLs are used in a variety of features. ACLs are made up of one or more access control entries (ACEs).

ACL Types

The ASA uses the following types of ACLs:

- Extended ACLs—Extended ACLs are the main type that you will use. These ACLs are used for access rules to permit and deny traffic through the device, and for traffic matching by many features, including service policies, AAA rules, WCCP, Botnet Traffic Filter, and VPN group and DAP policies. See Configure Extended ACLs, page 3-7.
- EtherType ACLs—EtherType ACLs apply to non-IP layer-2 traffic in transparent firewall mode. You can use these rules to permit or drop traffic based on the EtherType value in the layer-2 packet. With EtherType ACLs, you can control the flow of non-IP traffic across the device. See Configure EtherType ACLs, page 3-17.

- Webtype ACLs—Webtype ACLs are used for filtering clientless SSL VPN traffic. These ACLs can deny access based on URLs or destination addresses. See Configure Webtype ACLs, page 3-14.
- Standard ACLs—Standard ACLs identify traffic by destination address only. There are few features that use them: route maps and VPN filters. Because VPN filters also allow extended access lists, limit standard ACL use to route maps. See Configure Standard ACLs, page 3-13.

The following table lists some common uses for ACLs and the type to use.

Table 3-1 ACL Types and Common Uses

ACL Use	ACL Type	Description
Control network access for IP traffic (routed and transparent mode)	Extended	The ASA does not allow any traffic from a lower security interface to a higher security interface unless it is explicitly permitted by an extended ACL. **Note** To access the ASA interface for management access, you do not also need an ACL allowing the host IP address. You only need to configure management access according to the general operations configuration guide.
Identify traffic for AAA rules	Extended	AAA rules use ACLs to identify traffic.
Augment network access control for IP traffic for a given user	Extended, downloaded from a AAA server per user	You can configure the RADIUS server to download a dynamic ACL to be applied to the user, or the server can send the name of an ACL that you already configured on the ASA.
VPN access and filtering	Extended Standard	Group policies for remote access and site to site VPNs use standard or extended ACLs for filtering. Remote access VPNs also use extended ACLs for client firewall configurations and dynamic access policies.
Identify traffic in a traffic class map for Modular Policy Framework	Extended	ACLs can be used to identify traffic in a class map, which is used for features that support Modular Policy Framework. Features that support Modular Policy Framework include TCP and general connection settings, and inspection.
For transparent firewall mode, control network access for non-IP traffic	EtherType	You can configure an ACL that controls traffic based on its EtherType.
Identify route filtering and redistribution	Standard Extended	Various routing protocols use standard ACLs for route filtering and redistribution (through route maps) for IPv4 addresses, and extended ACLs for IPv6.
Filtering for clientless SSL VPN	Webtype	You can configure a webtype ACL to filter URLs and destinations.

ACL Names

Each ACL has a name or numeric ID, such as outside_in, OUTSIDE_IN, or 101. Limit the names to 241 characters or fewer.Consider using all uppercase letters to make it easier to find the name when viewing a running configuration.

Develop a naming convention that will help you identify the intended purpose of the ACL. For example, ASDM uses the convention *interface-name_purpose_direction*, such as "outside_access_in", for an ACL applied to the "outside" interface in the inbound direction.

Traditionally, ACL IDs were numbers. Standard ACLs were in the range 1-99 or 1300-1999. extended ACLs were in the range 100-199 or 2000-2699. The ASA does not enforce these ranges, but if you want to use numbers, you might want to stick to these conventions to maintain consistency with routers running IOS Software.

Access Control Entry Order

An ACL is made up of one or more ACEs. Unless you explicitly insert an ACE at a given line, each ACE that you enter for a given ACL name is appended to the end of the ACL.

The order of ACEs is important. When the ASA decides whether to forward or drop a packet, the ASA tests the packet against each ACE in the order in which the entries are listed. After a match is found, no more ACEs are checked.

Thus, if you place a more specific rule after a more general rule, the more specific rule might never be hit. For example, if you want to permit network 10.1.1.0/24, but drop traffic from host 10.1.1.15 on that subnet, the ACE that denies 10.1.1.15 must come before the one that permits 10.1.1.0/24. If the permit 10.1.1.0/24 ACE comes first, 10.1.1.15 will be allowed, and the deny ACE will never be matched.

In an extended ACL, use the **line** *number* parameter on the **access-list** command to insert rules at the right location. Use the **show access-list** *name* command to view the ACL entries and their line numbers to help determine the right number to use. For other types of ACL, you must rebuild the ACL (or better, use ASDM) to change the order of ACEs.

Permit/Deny vs. Match/Do Not Match

Access control entries either "permit" or "deny" traffic that matches the rule. When you apply an ACL to a feature that determines whether traffic is allowed through the ASA or is dropped, such as global and interface access rules, "permit" and "deny" mean what they say.

For other features, such as service policy rules, "permit" and "deny" actually mean "match" or "do not match." In these cases, the ACL is selecting the traffic that should receive the services of that feature, such as application inspection or redirection to a service module. "Denied" traffic is simply traffic that does not match the ACL, and thus will not receive the service.

Access Control Implicit Deny

All ACLs have an implicit deny statement at the end. Thus, for traffic controlling ACLs such as those applied to interfaces, if you do not explicitly permit a type of traffic, that traffic is dropped. For example, if you want to allow all users to access a network through the ASA except for one or more particular addresses, then you need to deny those particular addresses and then permit all others.

For ACLs used to select traffic for a service, you must explicitly "permit" the traffic; any traffic not "permitted" will not be matched for the service; "denied" traffic bypasses the service.

For EtherType ACLs, the implicit deny at the end of the ACL does not affect IP traffic or ARPs; for example, if you allow EtherType 8037, the implicit deny at the end of the ACL does not now block any IP traffic that you previously allowed with an extended ACL (or implicitly allowed from a high security

interface to a low security interface). However, if you *explicitly* deny all traffic with an EtherType ACE, then IP and ARP traffic is denied; only physical protocol traffic, such as auto-negotiation, is still allowed.

IP Addresses Used for Extended ACLs When You Use NAT

When you use NAT or PAT, you are translating addresses or ports, typically mapping between internal and external addresses. If you need to create an extended ACL that applies to addresses or ports that have been translated, you need to determine whether to use the real (untranslated) addresses or ports or the mapped ones. The requirement differs by feature.

Using the real address and port means that if the NAT configuration changes, you do not need to change the ACLs.

Features That Use Real IP Addresses

The following commands and features use real IP addresses in the ACLs, even if the address as seen on an interface is the mapped address:

- Access Rules (extended ACLs referenced by the **access-group** command)
- Service Policy Rules (Modular Policy Framework **match access-list** command)
- Botnet Traffic Filter traffic classification (**dynamic-filter enable classify-list** command)
- AAA Rules (**aaa ... match** commands)
- WCCP (**wccp redirect-list group-list** command)

For example, if you configure NAT for an inside server, 10.1.1.5, so that it has a publicly routable IP address on the outside, 209.165.201.5, then the access rule to allow the outside traffic to access the inside server needs to reference the server's real IP address (10.1.1.5), and not the mapped address (209.165.201.5).

```
hostname(config)# object network server1
hostname(config-network-object)# host 10.1.1.5
hostname(config-network-object)# nat (inside,outside) static 209.165.201.5

hostname(config)# access-list OUTSIDE extended permit tcp any host 10.1.1.5 eq www
hostname(config)# access-group OUTSIDE in interface outside
```

Features That Use Mapped IP Addresses

The following features use ACLs, but these ACLs use the mapped values as seen on an interface:

- IPsec ACLs
- **capture** command ACLs
- Per-user ACLs
- Routing protocol ACLs
- All other feature ACLs.

Time-Based ACEs

You can apply time range objects to extended and webtype ACEs so that the rules are active for specific time periods only. These types of rules let you differentiate between activity that is acceptable at certain times of the day but that is unacceptable at other times. For example, you could provide additional

restrictions during working hours, and relax them after work hours or at lunch. Conversely, you could essentially shut your network down during non-work hours. For information on creating time range objects, see Configure Time Ranges, page 2-9.

Note Users could experience a delay of approximately 80 to 100 seconds after the specified end time for the ACL to become inactive. For example, if the specified end time is 3:50, because the end time is inclusive, the command is picked up anywhere between 3:51:00 and 3:51:59. After the command is picked up, the ASA finishes any currently running task and then services the command to deactivate the ACL.

Guidelines for ACLs

Firewall Mode

- Extended and standard ACLs are supported in routed and transparent firewall modes.
- Webtype ACLs are supported in routed mode only.
- EtherType ACLs are supported in transparent mode only.

Failover and Clustering

Configuration sessions are not synchronized across failover or clustered units. When you commit the changes in a session, they are made in all failover and cluster units as normal.

IPv6

- Extended and webtype ACLs allow a mix of IPv4 and IPv6 addresses.
- Standard ACLs do not allow IPv6 addresses.
- EtherType ACLs do not contain IP addresses.

Additional Guidelines

- When you specify a network mask, the method is different from the Cisco IOS software **access-list** command. The ASA uses a network mask (for example, 255.255.255.0 for a Class C mask). The Cisco IOS mask uses wildcard bits (for example, 0.0.0.255).
- Normally, you cannot reference an object or object group that does not exist in an ACL or object group, or delete one that is currently referenced. You also cannot reference an ACL that does not exist in an **access-group** command (to apply access rules). However, you can change this default behavior so that you can "forward reference" objects or ACLs before you create them. Until you create the objects or ACLs, any rules or access groups that reference them are ignored. To enable forward referencing, use the **forward-reference enable** command.
- (Extended ACL only) Features That Do Not Support Identity Firewall, FQDN, and Cisco TrustSec ACLs—The following features use ACLs, but cannot accept an ACL with identity firewall (specifying user or group names), FQDN (fully-qualified domain names), or Cisco TrustSec values:
  - **route-map** command
  - VPN **crypto map** command
  - VPN **group-policy** command, except for **vpn-filter**
  - WCCP
  - DAP

Configure ACLs

The following sections explain how to configure the various types of ACL, Read the section on ACL basics to get the big picture, then the sections on specific types of ACL for the details.

Basic ACL Configuration and Management Options

An ACL is made up of one or more access control entries (ACEs) with the same ACL ID or name. To create a new ACL, you simply create an ACE with a new ACL name, and it becomes the first rule in the new ACL.

Working with an ACL, you can do the following things:

- **Examine the ACL contents and determine line numbers and hit counts**—Use the **show access-list** *name* command to view the contents of the ACL. Each row is an ACE, and includes the line number, which you will need to know if you want to insert new entries into an extended ACL. The information also includes a hit count for each ACE, which is how many times the rule was matched by traffic. For example:

```
hostname# show access-list outside_access_in
access-list outside_access_in; 3 elements; name hash: 0x6892a938
access-list outside_access_in line 1 extended permit ip 10.2.2.0 255.255.255.0 any
(hitcnt=0) 0xcc48b55c
access-list outside_access_in line 2 extended permit ip host
2001:DB8::0DB8:800:200C:417A any (hitcnt=0) 0x79797f94
access-list outside_access_in line 3 extended permit ip user-group LOCAL\\usergroup
any any (hitcnt=0) 0xb0f5b1e1
```

- **Add an ACE**—The command for adding an ACE is **access-list** *name* [**line** *line-num*] *type parameters*. The line number argument works for extended ACLs only. If you include the line number, the ACE is inserted at that location in the ACL, and the ACE that was at that location is moved down, along with the remainder of the ACEs (that is, inserting an ACE at a line number does not replace the old ACE at that line). If you do not include a line number, the ACE is added to the end of the ACL. The parameters available differ based on the ACL type; see the specific topics on each ACL type for details.
- **Add comments to an ACL (all types except webtype)**—Use the **access-list** *name* [**line** *line-num*] **remark** *text* command to add remarks into an ACL to help explain the purpose of an ACE. Best practice is to insert the remark before the ACE; if you view the configuration in ASDM, remarks will be associated with the ACE that follows the remarks. You can enter multiple remarks before an ACE to include an expanded comment. Each remark is limited to 100 characters. You can include leading spaces to help set off the remarks. If you do not include a line number, the remark is added to the end of the ACL. For example, you could add remarks before adding each ACE:

```
hostname(config)# access-list OUT remark - this is the inside admin address
hostname(config)# access-list OUT extended permit ip host 209.168.200.3 any
hostname(config)# access-list OUT remark - this is the hr admin address
hostname(config)# access-list OUT extended permit ip host 209.168.200.4 any
```

- **Edit or move an ACE or remark**—You cannot edit or move an ACE or remark. Instead, you must create a new ACE or remark with the desired values at the right location (using the line number), then delete the old ACE or remark. Because you can insert ACEs in extended ACLs only, you need to rebuild standard, webtype, or EtherType ACLs if you need to edit or move ACEs. It is far easier to reorganize a long ACL using ASDM.
- **Delete an ACE or remark**—Use the **no access-list** *parameters* command to remove an ACE or remark. Use the **show access-list** command to view the parameter string that you must enter: the string must exactly match an ACE or remark to delete it, with the exception of the **line** *line-num* argument, which is optional on the **no access-list** command.
- **Delete an entire ACL, including remarks**—Use the **clear configure access-list** *name* command. USE CAUTION! The command does not ask you for confirmation. If you do not include a name, every access list on the ASA is removed.
- **Rename an ACL**—Use the **access-list** *name* **rename** *new_name* command.
- **Apply the ACL to a policy**—Creating an ACL in and of itself does nothing to traffic. You must apply the ACL to a policy. For example, you can use the **access-group** command to apply an extended ACL to an interface, thus denying or permitting traffic that goes through the interface. For information on some of the uses of ACLs, see ACL Types, page 3-1.

Configure Extended ACLs

An extended ACL is composed of all ACEs with the same ACL ID or name. Extended ACLs are the most complex and feature-rich type of ACL, and you can use them for many features. The most noteworthy use of extended ACLs is as access groups applied globally or to interfaces, which determine the traffic that will be denied or permitted to flow through the box. But extended ACLs are also used to determine the traffic to which other services will be provided.

Because extended ACLs are complex, the following sections focus on creating ACEs to provide specific types of traffic matching. The first sections, on basic address-based ACEs and on TCP/UDP ACEs, build the foundation for the remaining sections.

Add an Extended ACE for IP Address or Fully-Qualified Domain Name-Based Matching

The basic extended ACE matches traffic based on source and destination addresses, including IPv4 and IPv6 addresses and fully-qualified domain names (FQDN), such as www.example.com. In fact, every type of extended ACE must include some specification for source and destination address, so this topic explains the minimum extended ACE.

Tip If you want to match traffic based on FQDN, you must create a network object for each FQDN.

To add an ACE for IP address or FQDN matching, use the following command:

```
access-list access_list_name [line line_number] extended {deny | permit}
protocol_argument source_address_argument dest_address_argument
[log [[level] [interval secs] | disable | default]]
[time-range time_range_name]
[inactive]
```

Example:

```
hostname(config)# access-list ACL_IN extended permit ip any any
```

The options are:

- *access_list_name*—The name of the new or existing ACL.
- Line number—The **line** *line_number* option specifies the line number at which insert the ACE; otherwise, the ACE is added to the end of the ACL.
- Permit or Deny—The **deny** keyword denies or exempts a packet if the conditions are matched. The **permit** keyword permits or includes a packet if the conditions are matched.
- Protocol—The *protocol_argument* specifies the IP protocol:
  - *name* or *number*—Specifies the protocol name or number. Specify **ip** to apply to all protocols.
  - **object-group** *protocol_grp_id*—Specifies a protocol object group created using the **object-group protocol** command. See Configure Service Objects and Service Groups, page 2-4.
  - **object** *service_obj_id*—Specifies a service object created using the **object service** command. A TCP, UDP, or ICMP service object can include a protocol and a source or destination port or ICMP type and code.
  - **object-group** *service_grp_id*—Specifies a service object group created using the **object-group service** command.
- Source Address, Destination Address—The *source_address_argument* specifies the IP address or FQDN from which the packet is being sent, and the *dest_address_argument* specifies the IP address or FQDN to which the packet is being sent:
  - **host** *ip_address*—Specifies an IPv4 host address.
  - *ip_address mask*—Specifies an IPv4 network address and subnet mask, such as 10.100.10.0 255.255.255.0.
  - *ipv6-address/prefix-length*—Specifies an IPv6 host or network address and prefix.
  - **any**, **any4**, and **any6**—**any** specifies both IPv4 and IPv6 traffic; **any4** specifies IPv4 traffic only; and **any6** specifies IPv6 traffic only.
  - **interface** *interface_name*—Specifies the name of an ASA interface. Use the interface name rather than IP address to match traffic based on which interface is the source or destination of the traffic.
  - **object** *nw_obj_id*—Specifies a network object created using the **object network** command. See Configure Network Objects and Groups, page 2-2.
  - **object-group** *nw_grp_id*—Specifies a network object group created using the **object-group network** command.
- Logging—**log** arguments set logging options when an ACE matches a packet for network access (an ACL applied with the **access-group** command). If you enter the **log** option without any arguments, you enable syslog message 106100 at the default level (6) and for the default interval (300 seconds). Log options are:

  - *level*—A severity level between 0 and 7. The default is 6 (informational). If you change this level for an active ACE, the new level applies to new connections; existing connections continue to be logged at the previous level.
  - **interval** *secs*—The time interval in seconds between syslog messages, from 1 to 600. The default is 300. This value is also used as the timeout value for deleting an inactive flow from the cache used to collect drop statistics.
  - **disable**—Disables all ACE logging.
  - **default**—Enables logging to message 106023 for denied packets. This setting is the same as not including the **log** option.
- Time Range—The **time-range** *time_range_name* option specifies a time range object, which determines the times of day and days of the week in which the ACE is active. If you do not include a time range, the ACE is always active.
- Activation—Use the **inactive** option to disable the ACE without deleting it. To reenable it, enter the entire ACE without the inactive keyword.

Add an Extended ACE for TCP or UDP-Based Matching, with Ports

The TCP/UDP extended ACE is just the basic address-matching ACE where the protocol is **tcp** or **udp**. Because these protocols use ports, you can add port specifications to the ACE. For example, you can target HTTP traffic on TCP port 80.

To add an ACE for IP address or FQDN matching, where the protocol is TCP or UDP, use the following command:

```
access-list access_list_name [line line_number] extended {deny | permit}
{tcp | udp} source_address_argument [port_argument] dest_address_argument [port_argument]
[log [[level] [interval secs] | disable | default]]
[time-range time_range_name]
[inactive]
```

Example:

```
hostname(config)# access-list ACL_IN extended deny tcp any host 209.165.201.29 eq www
```

The *port_argument* option specifies the source or destination port. If you do not specify ports, all ports are matched. Available arguments include:

- *operator port*—The *operator* can be one of the following:
  - **lt**—less than
  - **gt**—greater than
  - **eq**—equal to
  - **neq**—not equal to
  - **range**—an inclusive range of values. When you use this operator, specify two port numbers, for example:

    ```
    range 100 200
    ```

  The *port* can be the integer or name of a TCP or UDP port. DNS, Discard, Echo, Ident, NTP, RPC, SUNRPC, and Talk each require one definition for TCP and one for UDP. TACACS+ requires one definition for port 49 on TCP.
- **object** *service_obj_id*—Specifies a service object created using the **object service** command. See Configure Service Objects and Service Groups, page 2-4.

- **object-group** *service_grp_id*—Specifies a service object group created using the **object-group service** command.

For an explanation of the other keywords, see Add an Extended ACE for IP Address or Fully-Qualified Domain Name-Based Matching, page 3-7.

Add an Extended ACE for ICMP-Based Matching

The ICMP extended ACE is just the basic address-matching ACE where the protocol is **icmp** or **icmp6**. Because these protocols have type and code values, you can add type and code specifications to the ACE. For example, you can target ICMP Echo Request traffic (pings).

To add an ACE for IP address or FQDN matching, where the protocol is ICMP or ICMP6, use the following command:

```
access-list access_list_name [line line_number] extended {deny | permit}
{icmp | icmp6} source_address_argument dest_address_argument [icmp_argument]
[log [[level] [interval secs] | disable | default]]
[time-range time_range_name]
[inactive]
```

Example:

```
hostname(config)# access-list abc extended permit icmp any any object-group obj_icmp_1
hostname(config)# access-list abc extended permit icmp any any echo
```

The *icmp_argument* option specifies the ICMP type and code.

- *icmp_type* [*icmp_code*]—Specifies the ICMP type by name or number, and the optional ICMP code for that type. If you do not specify the code, then all codes are used.
- **object-group** *icmp_grp_id*—Specifies an object group for ICMP/ICMP6 created using the **object-group service** or (deprecated) **object-group icmp** command.

For an explanation of the other keywords, see Add an Extended ACE for IP Address or Fully-Qualified Domain Name-Based Matching, page 3-7.

Add an Extended ACE for User-Based Matching (Identity Firewall)

The user-based extended ACE is just the basic address-matching ACE where you include username or user group to the source matching criteria. By creating rules based on user identity, you can avoid tying rules to static host or network addresses. For example, if you define a rule for user1, and the identity firewall feature maps that user to a host assigned 10.100.10.3 one day, but 192.168.1.5 the next day, the user-based rule still applies.

Because you must still supply source and destination addresses, broaden the source address to include the likely addresses that will be assigned to the user (normally through DHCP). For example, user “LOCAL\user1 any” will match the LOCAL\user1 user no matter what address is assigned, whereas “LOCAL\user1 10.100.1.0 255.255.255.0” matches the user only if the address is on the 10.100.1.0/24 network.

By using group names, you can define rules based on entire classes of users, such as students, teachers, managers, engineers, and so forth.

To add an ACE for user or group matching, use the following command:

```
access-list access_list_name [line line_number] extended {deny | permit} protocol_argument
[user_argument] source_address_argument [port_argument]
dest_address_argument [port_argument]
```

```
[log [[level] [interval secs] | disable | default]]
[time-range time_range_name]
[inactive]
```

Example:

```
hostname(config)# access-list v1 extended permit ip user LOCAL\idfw
any 10.0.0.0 255.255.255.0
```

The *user_argument* option specifies the user or group for which to match traffic in addition to the source address. Available arguments include the following:

- **object-group-user** *user_obj_grp_id*—Specifies a user object group created using the **object-group user** command.
- **user** {[*domain_nickname*\]*name* | **any** | **none**}—Specifies a username. Specify **any** to match all users with user credentials, or **none** to match addresses that are not mapped to usernames. These options are especially useful for combining **access-group** and **aaa authentication match** policies.
- **user-group** [*domain_nickname*\\]*user_group_name*—Specifies a user group name. Note the double \\ separating the domain and group name.

For an explanation of the other keywords, see Add an Extended ACE for IP Address or Fully-Qualified Domain Name-Based Matching, page 3-7.

Tip You can include both user and Cisco Trustsec security groups in a given ACE. See Add an Extended ACE for Security Group-Based Matching (Cisco TrustSec), page 3-11.

Add an Extended ACE for Security Group-Based Matching (Cisco TrustSec)

The security group (Cisco TrustSec) extended ACE is just the basic address-matching ACE where you include security groups or tags to the source or destination matching criteria. By creating rules based on security groups, you can avoid tying rules to static host or network addresses. Because you must still supply source and destination addresses, broaden the addresses to include the likely addresses that will be assigned to users (normally through DHCP).

Tip Before adding this type of ACE, configure Cisco TrustSec as described in Chapter 6, "ASA and Cisco TrustSec."

To add an ACE for security group matching, use the following command:

```
access-list access_list_name [line line_number] extended {deny | permit} protocol_argument
[security_group_argument] source_address_argument [port_argument]
[security_group_argument] dest_address_argument [port_argument] [log [[level]
[interval secs] | disable | default]] [inactive | time-range time_range_name]
```

Example:

```
hostname(config)# access-list INSIDE_IN extended permit ip
security-group name my-group any any
```

The *security_group_argument* option specifies the security group for which to match traffic in addition to the source or destination address. Available arguments include the following:

- **object-group-security** *security_obj_grp_id*—Specifies a security object group created using the **object-group security** command.

- **security-group** {**name** *security_grp_id* | **tag** *security_grp_tag*}—Specifies a security group name or tag.

For an explanation of the other keywords, see Add an Extended ACE for IP Address or Fully-Qualified Domain Name-Based Matching, page 3-7.

Tip You can include both user and Cisco Trustsec security groups in a given ACE. See Add an Extended ACE for User-Based Matching (Identity Firewall), page 3-10.

Examples for Extended ACLs

The following ACL allows all hosts (on the interface to which you apply the ACL) to go through the ASA:

```
hostname(config)# access-list ACL_IN extended permit ip any any
```

The following ACL prevents hosts on 192.168.1.0/24 from accessing the 209.165.201.0/27 network for TCP-based traffic. All other addresses are permitted.

```
hostname(config)# access-list ACL_IN extended deny tcp 192.168.1.0 255.255.255.0
209.165.201.0 255.255.255.224
hostname(config)# access-list ACL_IN extended permit ip any any
```

If you want to restrict access to selected hosts only, then enter a limited permit ACE. By default, all other traffic is denied unless explicitly permitted.

```
hostname(config)# access-list ACL_IN extended permit ip 192.168.1.0 255.255.255.0
209.165.201.0 255.255.255.224
```

The following ACL restricts all hosts (on the interface to which you apply the ACL) from accessing a website at address 209.165.201.29. All other traffic is allowed.

```
hostname(config)# access-list ACL_IN extended deny tcp any host 209.165.201.29 eq www
hostname(config)# access-list ACL_IN extended permit ip any any
```

The following ACL that uses object groups restricts several hosts on the inside network from accessing several web servers. All other traffic is allowed.

```
hostname(config-network)# access-list ACL_IN extended deny tcp object-group denied
object-group web eq www
hostname(config)# access-list ACL_IN extended permit ip any any
hostname(config)# access-group ACL_IN in interface inside
```

The following example temporarily disables an ACL that permits traffic from one group of network objects (A) to another group of network objects (B):

```
hostname(config)# access-list 104 permit ip host object-group A object-group B inactive
```

To implement a time-based ACE, use the **time-range** command to define specific times of the day and week. Then use the **access-list extended** command to bind the time range to an ACE. The following example binds an ACE in the "Sales" ACL to a time range named "New_York_Minute."

```
hostname(config)# access-list Sales line 1 extended deny tcp host 209.165.200.225 host
209.165.201.1 time-range New_York_Minute
```

The following example shows a mixed IPv4/IPv6 ACL:

```
hostname(config)# access-list demoacl extended permit ip 2001:DB8:1::/64 10.2.2.0
255.255.255.0
hostname(config)# access-list demoacl extended permit ip 2001:DB8:1::/64 2001:DB8:2::/64
hostname(config)# access-list demoacl extended permit ip host 10.3.3.3 host 10.4.4.4
```

Example of Converting Addresses to Objects for Extended ACLs

The following normal ACL that does not use object groups restricts several hosts on the inside network from accessing several web servers. All other traffic is allowed.

```
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.4 host 209.165.201.29
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.78 host 209.165.201.29
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.89 host 209.165.201.29
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.4 host 209.165.201.16
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.78 host 209.165.201.16
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.89 host 209.165.201.16
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.4 host 209.165.201.78
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.78 host 209.165.201.78
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.89 host 209.165.201.78
eq www
hostname(config)# access-list ACL_IN extended permit ip any any
hostname(config)# access-group ACL_IN in interface inside
```

If you make two network object groups, one for the inside hosts, and one for the web servers, then the configuration can be simplified and can be easily modified to add more hosts:

```
hostname(config)# object-group network denied
hostname(config-network)# network-object host 10.1.1.4
hostname(config-network)# network-object host 10.1.1.78
hostname(config-network)# network-object host 10.1.1.89

hostname(config-network)# object-group network web
hostname(config-network)# network-object host 209.165.201.29
hostname(config-network)# network-object host 209.165.201.16
hostname(config-network)# network-object host 209.165.201.78

hostname(config-network)# access-list ACL_IN extended deny tcp object-group denied
object-group web eq www
hostname(config)# access-list ACL_IN extended permit ip any any
hostname(config)# access-group ACL_IN in interface inside
```

Configure Standard ACLs

A standard ACL is composed of all ACEs with the same ACL ID or name. Standard ACLs are used for a limited number of features, such as route maps or VPN filters. A standard ACL uses IPv4 addresses only, and defines destination addresses only.

To add a standard access list entry, use the following command:

```
hostname(config)# access-list access_list_name standard {deny | permit}
{any4 | host ip_address | ip_address mask}
```

Example:

```
hostname(config)# access-list OSPF standard permit 192.168.1.0 255.255.255.0
```

The options are:

- Name—The *access_list_name* argument specifies the name of number of an ACL. Traditional numbers for standard ACLs are 1-99 or 1300-1999, but you can use any name or number. You create a new ACL if the ACL does not already exist, otherwise, you are adding the entry to the end of the ACL.
- Permit or Deny—The **deny** keyword denies or exempts a packet if the conditions are matched. The **permit** keyword permits or includes a packet if the conditions are matched.
- Destination Address—The **any4** keyword matches all IPv4 addresses. The **host** *ip_address* argument matches a host IPv4 address. The *ip_address ip_mask* argument matches an IPv4 subnet, for example, 10.1.1.0 255.255.255.0.

Configure Webtype ACLs

Webtype ACLs are used for filtering clientless SSL VPN traffic, constraining user access to specific networks, subnets, hosts, and Web servers. If you do not define a filter, all connections are allowed. A webtype ACL is composed of all ACEs with the same ACL ID or name.

With webtype ACLs, you can match traffic based on URLs or destination addresses. A single ACE cannot mix these specifications. The following sections explain each type of ACE.

- Add a Webtype ACE for URL Matching, page 3-14
- Adding a Webtype ACE for IP Address Matching, page 3-15
- Examples for Webtype ACLs, page 3-16

Add a Webtype ACE for URL Matching

To match traffic based on the URL the user is trying to access, use the following command;

```
access-list access_list_name webtype {deny | permit} url {url_string | any}
[log [[level] [interval secs] | disable | default]]
[time_range time_range_name]]
[inactive]
```

Example:

```
hostname(config)# access-list acl_company webtype deny url http://*.example.com
```

The options are:

- *access_list_name*—The name of the new or existing ACL. If the ACL already exists, you are adding the ACE to the end of the ACL.
- Permit or Deny—The **deny** keyword denies or exempts a packet if the conditions are matched. The **permit** keyword permits or includes a packet if the conditions are matched.
- URL—The **url** keyword specifies the URL to match. Use **url any** to match all URL-based traffic. Otherwise, enter a URL string, which can include wildcards. Following are some tips and limitations on specifying URLs:
  - Specify **any** to match all URLs.
  - ‘Permit url any' will allow all the URLs that have the format protocol://server-ip/path and will block traffic that does not match this pattern, such as port-forwarding. There should be an ACE to allow connections to the required port (port 1494 in the case of Citrix) so that an implicit deny does not occur.

  - Smart tunnel and ica plug-ins are not affected by an ACL with ‘permit url any’ because they match smart-tunnel:// and ica:// types only.
  - You can use these protocols: cifs://, citrix://, citrixs://, ftp://, http://, https://, imap4://, nfs://, pop3://, smart-tunnel://, and smtp://. You can also use wildcards in the protocol; for example, htt* matches http and https, and an asterisk * matches all protocols. For example, *://*.example.com matches any type URL-based traffic to the example.com network.
  - If you specify a smart-tunnel:// URL, you can include the server name only. The URL cannot contain a path. For example, smart-tunnel://www.example.com is acceptable, but smart-tunnel://www.example.com/index.html is not.
  - An asterisk * matches none or any number of characters. To match any http URL, enter http://*/*.
  - A question mark ? matches any one character exactly.
  - Square brackets [] are range operators, matching any character in the range. For example, to match both http://www.cisco.com:80/ and http://www.cisco.com:81/, enter **http://www.cisco.com:8[01]/**.

- Logging—**log** arguments set logging options when an ACE matches a packet. If you enter the **log** option without any arguments, you enable syslog message 106102 at the default level (6) and for the default interval (300 seconds). Log options are:
  - *level*—A severity level between 0 and 7. The default is 6.
  - **interval** *secs*—The time interval in seconds between syslog messages, from 1 to 600. The default is 300.
  - **disable**—Disables all ACL logging.
  - **default**—Enables logging to message 106103. This setting is the same as not including the **log** option.
- Time Range—The **time-range** *time_range_name* option specifies a time range object, which determines the times of day and days of the week in which the ACE is active. If you do not include a time range, the ACE is always active.
- Activation—Use the **inactive** option to disable the ACE without deleting it. To reenable it, enter the entire ACE without the inactive keyword.

Adding a Webtype ACE for IP Address Matching

You can match traffic based on the destination address the user is trying to access. The webtype ACL can include a mix of IPv4 and IPv6 addresses in addition to URL specifications.

To add a webtype ACE for IP address matching, use the following command:

```
access-list access_list_name webtype {deny | permit}
tcp dest_address_argument [operator port]
[log [[level] [interval secs] | disable | default]]
[time_range time_range_name]]
[inactive]]
```

Example:

```
hostname(config)# access-list acl_company webtype permit tcp any
```

For an explanation of keywords not explained here, see Add a Webtype ACE for URL Matching, page 3-14. Keywords and arguments specific to this type of ACE include the following:

- **tcp**—The TCP protocol. Webtype ACLs match TCP traffic only.
- Destination Address—The *dest_address_argument* specifies the IP address to which the packet is being sent:
  - **host** *ip_address*—Specifies an IPv4 host address.
  - *dest_ip_address mask*—Specifies an IPv4 network address and subnet mask, such as 10.100.10.0 255.255.255.0.
  - *ipv6-address/prefix-length*—Specifies an IPv6 host or network address and prefix.
  - **any**, **any4**, and **any6**—**any** specifies both IPv4 and IPv6 traffic; **any4** specifies IPv4 traffic only; and **any6** specifies IPv6 traffic only.
- *operator port*—The destination port. If you do not specify ports, all ports are matched. The *operator* can be one of the following:
  - **lt**—less than
  - **gt**—greater than
  - **eq**—equal to
  - **neq**—not equal to
  - **range**—an inclusive range of values. When you use this operator, specify two port numbers, for example:

    ```
    range 100 200
    ```

  The *port* can be the integer or name of a TCP port.

Examples for Webtype ACLs

The following example shows how to deny access to a specific company URL:

```
hostname(config)# access-list acl_company webtype deny url http://*.example.com
```

The following example shows how to deny access to a specific web page:

```
hostname(config)# access-list acl_file webtype deny url
https://www.example.com/dir/file.html
```

The following example shows how to deny HTTP access to any URL on a specific server through port 8080:

```
hostname(config)# access-list acl_company webtype deny url http://my-server:8080/*
```

The following examples show how to use wildcards in webtype ACLs.

- The following example matches URLs such as http://www.example.com/layouts/1033:

  ```
  access-list VPN-Group webtype permit url http://www.example.com/*
  ```

- The following example matches URLs such as http://www.example.com/ and http://www.example.net/:

  access-list test webtype permit url http://www.example.*

- The following example matches URLs such as http://www.example.com and ftp://wwz.example.com:

```
access-list test webtype permit url *://ww?.e*co*/
```

- The following example matches URLs such as http://www.cisco.com:80 and https://www.cisco.com:81:

```
access-list test webtype permit url *://ww?.c*co*:8[01]/
```

  The range operator "[]" in the preceding example specifies that either character **0** or **1** can occur at that location.

- The following example matches URLs such as http://www.example.com and http://www.example.net:

```
access-list test webtype permit url http://www.[a-z]xample?*/
```

  The range operator "[]" in the preceding example specifies that any character in the range from **a** to **z** can occur.

- The following example matches http or https URLs that include "cgi" somewhere in the file name or path.

```
access-list test webtype permit url htt*://*/*cgi?*
```

Note To match any http URL, you must enter **http://*/*** instead of http://*.

The following example shows how to enforce a webtype ACL to disable access to specific CIFS shares.

In this scenario we have a root folder named "shares" that contains two sub-folders named "Marketing_Reports" and "Sales_Reports." We want to specifically deny access to the "shares/Marketing_Reports" folder.

```
access-list CIFS_Avoid webtype deny url cifs://172.16.10.40/shares/Marketing_Reports.
```

However, due to the implicit "deny all" at the end of the ACL, the above ACL makes all of the sub-folders inaccessible ("shares/Sales_Reports" and "shares/Marketing_Reports"), including the root folder ("shares").

To fix the problem, add a new ACL to allow access to the root folder and the remaining sub-folders:

```
access-list CIFS_Allow webtype permit url cifs://172.16.10.40/shares*
```

Configure EtherType ACLs

EtherType ACLs apply to non-IP layer-2 traffic in transparent firewall mode. You can use these rules to permit or drop traffic based on the EtherType value in the layer-2 packet. With EtherType ACLs, you can control the flow of non-IP traffic across the ASA. Note that 802.3-formatted frames are not handled by the ACL because they use a length field as opposed to a type field.

To add an EtherType ACE, use the following command:

access-list *access_list_name* **ethertype** {**deny** | **permit**}
{**ipx** | **bpdu** | **mpls-unicast** | **mpls-multicast** | **isis** | **any** | *hex_number*}

Example:

```
hostname(config)# access-list ETHER ethertype deny ipx
```

The options are:

- *access_list_name*—The name of the new or existing ACL. If the ACL already exists, you are adding the ACE to the end of the ACL.
- Permit or Deny—The **deny** keyword denies a packet if the conditions are matched. The **permit** keyword permits a packet if the conditions are matched.
- Traffic Matching Criteria—You can match traffic using the following options:
  - **ipx**—Internet Packet Exchange (IPX).
  - **bpdu**—bridge protocol data units, which are allowed by default.
  - **mpls-multicast**— MPLS multicast.
  - **mpls-unicast**—MPLS unicast.
  - **isis**—Intermediate System to Intermediate System (IS-IS).
  - **any**—Matches all traffic.
  - *hex_number*—Any EtherType that can be identified by a 16-bit hexadecimal number 0x600 to 0xffff. See RFC 1700, "Assigned Numbers," at http://www.ietf.org/rfc/rfc1700.txt for a list of EtherTypes.

Examples for EtherType ACLs

The following examples shows how to configure EtherType ACLs, including how to apply them to an interface.

The following sample ACL allows common traffic originating on the inside interface:

```
hostname(config)# access-list ETHER ethertype permit ipx
hostname(config)# access-list ETHER ethertype permit mpls-unicast
hostname(config)# access-group ETHER in interface inside
```

The following ACL allows some EtherTypes through the ASA, but it denies IPX:

```
hostname(config)# access-list ETHER ethertype deny ipx
hostname(config)# access-list ETHER ethertype permit 1234
hostname(config)# access-list ETHER ethertype permit mpls-unicast
hostname(config)# access-group ETHER in interface inside
hostname(config)# access-group ETHER in interface outside
```

The following ACL denies traffic with EtherType 0x1256, but it allows all others on both interfaces:

```
hostname(config)# access-list nonIP ethertype deny 1256
hostname(config)# access-list nonIP ethertype permit any
hostname(config)# access-group ETHER in interface inside
hostname(config)# access-group ETHER in interface outside
```

Edit ACLs in an Isolated Configuration Session

When you edit an ACL used for access rules or any other purpose, the change is immediately implemented and impacts traffic. With access rules, you can enable the transactional commit model to ensure that new rules become active only after rule compilation is complete, but the compilation happens after each ACE you edit.

If you want to further isolate the impact of editing ACLs, you can make your changes in a "configuration session," which is an isolated mode that allows you to edit several ACEs and objects before explicitly committing your changes. Thus, you can ensure that all of your intended changes are complete before you change device behavior.

Before You Begin

- You can edit ACLs that are referenced by an access-group command, but you cannot edit ACLs that are referenced by any other command. You can also edit unreferenced ACLs or create new ones.
- You can create or edit objects and object groups, but if you create one in a session, you cannot edit it in the same session. If the object is not defined as desired, you must commit your changes and then edit the object, or discard the entire session and start over.
- When you edit an ACL that is referenced by an access-group command (access rules), the transactional commit model is used when you commit the session. Thus, the ACL is completely compiled before the new ACL replaces the old version.
- If you enable forward referencing of ACL and object names (the **forward-reference enable** command), you can delete an ACL that is referenced by an access-group command (access rules), and then recreate the ACL. When you commit changes, the new version of the ACL will be used after compilation is complete. You can also create rules that refer to objects that do not exist, or delete objects that are in use by access rules. However, you will get a commit error if you delete an object used by other rules, such as NAT.

Procedure

Step 1 Start the session.

```
hostname#configure session session_name
hostname(config-s)#
```

If the *session_name* already exists, you open that session. Otherwise, you are creating a new session.

Use the **show configuration session** command to view the existing sessions. You can have at most 3 sessions active at a time. If you need to delete an old unused session, use the **clear configuration session** *session_name* command.

If you cannot open an existing session because someone else is editing it, you can clear the flag that indicates the session is being edited. Do this only if you are certain the session is not actually being edited. Use the **clear session** *session_name* **access** command to reset the flag.

Step 2 (Uncommitted sessions only.) Make your changes. You can use the following basic commands with any of their parameters:

- **access-list**
- **object**
- **object-group**

Step 3 Decide what to do with the session. The commands available depend on whether you have previously committed the session. Possible commands are:

- **exit**—To simply exit the session without committing or discarding changes, so that you can return later.
- **commit** [**noconfirm** [**revert-save** | **config-save**]]—(Uncommitted sessions only.) To commit your changes. You are asked if you want to save the session. You can save the revert session (**revert-save**), which lets you undo your changes using the **revert** command, or the configuration session (**config-save**), which includes all of the changes made in the session (allowing you to commit the

same changes again if you would like to). If you save the revert or configuration session, the changes are committed, but the session remains active. You can open the session and revert or recommit the changes. You can avoid the prompt by including the **noconfirm** option and optionally, the desired save option.

- **abort**—(Uncommitted sessions only.) To abandon your changes and delete the session. If you want to keep the session, exit the session and use the **clear session** *session_name* **configuration** command, which empties the session without deleting it.
- **revert**—(Committed sessions only.) To undo your changes, returning the configuration back to what it was before you committed the session, and delete the session.
- **show configuration session** [*session_name*]—To show the changes made in the session.

Monitoring ACLs

To monitor ACLs, enter one of the following commands:

Command	Purpose
show access-list [*name*]	Displays the access lists, including the line number for each ACE and hit counts. Include an ACL name or you will see all access lists.
show running-config access-list [*name*]	Displays the current running access-list configuration. Include an ACL name or you will see all access lists.

History for ACLs

Feature Name	Releases	Description
Extended, standard, webtype ACLs	7.0(1)	ACLs are used to control network access or to specify traffic for many features to act upon. An extended access control list is used for through-the-box access control and several other features. Standard ACLs are used in route maps and VPN filters. Webtype ACLs are used in clientless SSL VPN filtering. EtherType ACLs control non-IP layer 2 traffic. We introduced the following commands: **access-list extended, access-list standard, access-list webtype**, **access-list ethertype**.
Real IP addresses in extended ACLs	8.3(1)	When using NAT or PAT, mapped addresses and ports are no longer used in an ACL for several features. You must use the real, untranslated addresses and ports for these features. Using the real address and port means that if the NAT configuration changes, you do not need to change the ACLs. For more information, see IP Addresses Used for Extended ACLs When You Use NAT, page 3-4.
Support for Identity Firewall in extended ACLs	8.4(2)	You can now use identity firewall users and groups for the source and destination. You can use an identity firewall ACL with access rules, AAA rules, and for VPN authentication. We modified the following commands: **access-list extended**.
EtherType ACL support for IS-IS traffic	8.4(5), 9.1(2)	In transparent firewall mode, the ASA can now control IS-IS traffic using an EtherType ACL. We modified the following command: **access-list ethertype** {**permit** | **deny**} **isis**.
Support for Cisco TrustSec in extended ACLs	9.0(1)	You can now use Cisco TrustSec security groups for the source and destination. You can use an identity firewall ACL with access rules. We modified the following commands: **access-list extended**.

Feature Name	Releases	Description
Unified extended and webtype ACLs for IPv4 and IPv6	9.0(1)	Extended and webtype ACLs now support IPv4 and IPv6 addresses. You can even specify a mix of IPv4 and IPv6 addresses for the source and destination. The **any** keyword was changed to represent IPv4 and IPv6 traffic. The **any4** and **any6** keywords were added to represent IPv4-only and IPv6-only traffic, respectively. The IPv6-specific ACLs are deprecated. Existing IPv6 ACLs are migrated to extended ACLs. See the release notes for more information about migration. We modified the following commands: **access-list extended**, **access-list webtype**. We removed the following commands: **ipv6 access-list**, **ipv6 access-list webtype**, **ipv6-vpn-filter**.
Extended ACL and object enhancement to filter ICMP traffic by ICMP code	9.0(1)	ICMP traffic can now be permitted/denied based on ICMP code. We introduced or modified the following commands: **access-list extended**, **service-object**, **service**.
Configuration session for editing ACLs and objects. Forward referencing of objects and ACLs in access rules.	9.3(2)	You can now edit ACLs and objects in an isolated configuration session. You can also forward reference objects and ACLs, that is, configure rules and access groups for objects or ACLs that do not yet exist. We introduced the **clear configuration session**, **clear session**, **configure session**, **forward-reference**, and **show configuration session** commands.

Access Rules

This chapter describes how to control network access through or to the ASA using access rules. You use access rules to control network access in both routed and transparent firewall modes. In transparent mode, you can use both access rules (for Layer 3 traffic) and EtherType rules (for Layer 2 traffic).

Note To access the ASA interface for management access, you do not also need an access rule allowing the host IP address. You only need to configure management access according to the general operations configuration guide.

Controlling Network Access

Access rules determine which traffic is allowed through the ASA. There are several different layers of rules that work together to implement your access control policy:

- Extended access rules (Layer 3+ traffic) assigned to interfaces—You can apply separate rule sets (ACLs) in the inbound and outbound directions. An extended access rule permits or denies traffic based on the source and destination traffic criteria.
- Extended access rules assigned globally—You can create a single global rule set, which serves as your default access control. The global rules are applied after interface rules.
- Management access rules (Layer 3+ traffic)—You can apply a single rule set to cover traffic directed at an interface, which would typically be management traffic. In the CLI, these are “control plane” access groups. For ICMP traffic directed at the device, you can alternatively configure ICMP rules.
- EtherType rules (Layer 2 traffic) assigned to interfaces (transparent firewall mode only)—You can apply separate rule sets in the inbound and outbound directions. EtherType rules control network access for non-IP traffic. An EtherType rule permits or denies traffic based on the EtherType.

In transparent firewall mode, you can combine extended access rules, management access rules, and EtherType rules on the same interface.

General Information About Rules

This section describes information for both access rules and EtherType rules, and it includes the following topics:

Interface Access Rules and Global Access Rules

You can apply an access rule to a specific interface, or you can apply an access rule globally to all interfaces. You can configure global access rules in conjunction with interface access rules, in which case, the specific inbound interface access rules are always processed before the general global access rules. Global access rules apply only to inbound traffic.

Inbound and Outbound Rules

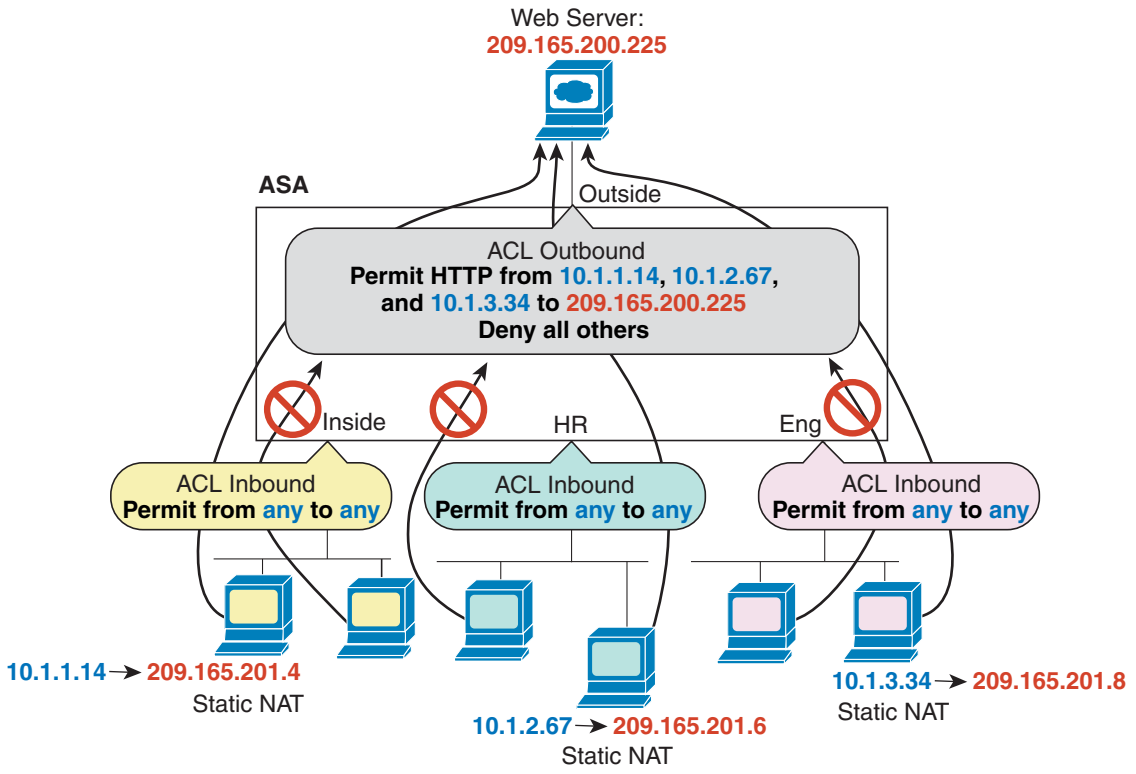
You can configure access rules based on the direction of traffic:

- Inbound—Inbound access rules apply to traffic as it enters an interface. Global and management access rules are always inbound.
- Outbound—Outbound rules apply to traffic as it exits an interface.

Note "Inbound" and "outbound" refer to the application of an ACL on an interface, either to traffic entering the ASA on an interface or traffic exiting the ASA on an interface. These terms do not refer to the movement of traffic from a lower security interface to a higher security interface, commonly known as inbound, or from a higher to lower interface, commonly known as outbound.

An outbound ACL is useful, for example, if you want to allow only certain hosts on the inside networks to access a web server on the outside network. Rather than creating multiple inbound ACLs to restrict access, you can create a single outbound ACL that allows only the specified hosts. (See the following figure.) The outbound ACL prevents any other hosts from reaching the outside network.

Figure 4-1 Outbound ACL



See the following commands for this example:

```
hostname(config)# access-list OUTSIDE extended permit tcp host 10.1.1.14
host 209.165.200.225 eq www
hostname(config)# access-list OUTSIDE extended permit tcp host 10.1.2.67
host 209.165.200.225 eq www
hostname(config)# access-list OUTSIDE extended permit tcp host 10.1.3.34
host 209.165.200.225 eq www
hostname(config)# access-group OUTSIDE out interface outside
```

Rule Order

The order of rules is important. When the ASA decides whether to forward or drop a packet, the ASA tests the packet against each rule in the order in which the rules are listed in the applied ACL. After a match is found, no more rules are checked. For example, if you create an access rule at the beginning that explicitly permits all traffic for an interface, no further rules are ever checked.

Implicit Permits

For routed mode, the following types of traffic are allowed through by default:

- Unicast IPv4 and IPv6 traffic from a higher security interface to a lower security interface.

For transparent mode, the following types of traffic are allowed through by default:

- Unicast IPv4 and IPv6 traffic from a higher security interface to a lower security interface.
- ARPs in both directions. (You can control ARP traffic using ARP inspection, but you cannot control it by access rule.)
- BPDUs in both directions.

For other traffic, you need to use either an extended access rule (IPv4 and IPv6) or an EtherType rule (non-IP).

Implicit Deny

ACLs have an implicit deny at the end of the list, so unless you explicitly permit it, traffic cannot pass. For example, if you want to allow all users to access a network through the ASA except for particular addresses, then you need to deny the particular addresses and then permit all others.

For EtherType ACLs, the implicit deny at the end of the ACL does not affect IP traffic or ARPs; for example, if you allow EtherType 8037, the implicit deny at the end of the ACL does not now block any IP traffic that you previously allowed with an extended ACL (or implicitly allowed from a high security interface to a low security interface). However, if you explicitly deny all traffic with an EtherType rule, then IP and ARP traffic is denied; only physical protocol traffic, such as auto-negotiation, is still allowed.

If you configure a global access rule, then the implicit deny comes *after* the global rule is processed. See the following order of operations:

1. Interface access rule.
2. Global access rule.
3. Implicit deny.

NAT and Access Rules

Access rules always use the real IP addresses when determining an access rule match, even if you configure NAT. For example, if you configure NAT for an inside server, 10.1.1.5, so that it has a publicly routable IP address on the outside, 209.165.201.5, then the access rule to allow the outside traffic to access the inside server needs to reference the server's real IP address (10.1.1.5), and not the mapped address (209.165.201.5).

Extended Access Rules

This section describes information about extended access rules.

Extended Access Rules for Returning Traffic

For TCP and UDP connections for both routed and transparent mode, you do not need an access rule to allow returning traffic because the ASA allows all returning traffic for established, bidirectional connections.

For connectionless protocols such as ICMP, however, the ASA establishes unidirectional sessions, so you either need access rules to allow ICMP in both directions (by applying ACLs to the source and destination interfaces), or you need to enable the ICMP inspection engine. The ICMP inspection engine treats ICMP sessions as bidirectional connections. To control ping, specify **echo-reply** (**0**) (ASA to host) or **echo** (**8**) (host to ASA).

Allowing Broadcast and Multicast Traffic through the Transparent Firewall Using Access Rules

In routed firewall mode, broadcast and multicast traffic is blocked even if you allow it in an access rule, including unsupported dynamic routing protocols and DHCP (unless you configure DHCP relay). Transparent firewall mode can allow any IP traffic through.

Note Because these special types of traffic are connectionless, you need to apply an access rule to both interfaces, so returning traffic is allowed through.

The following table lists common traffic types that you can allow through the transparent firewall.

Table 4-1 Transparent Firewall Special Traffic

Traffic Type	Protocol or Port	Notes
DHCP	UDP ports 67 and 68	If you enable the DHCP server, then the ASA does not pass DHCP packets.
EIGRP	Protocol 88	—
OSPF	Protocol 89	—
Multicast streams	The UDP ports vary depending on the application.	Multicast streams are always destined to a Class D address (224.0.0.0 to 239.x.x.x).
RIP (v1 or v2)	UDP port 520	—

Management Access Rules

You can configure access rules that control management traffic destined to the ASA. Access control rules for to-the-box management traffic (defined by such commands as **http**, **ssh**, or **telnet**) have higher precedence than a management access rule applied with the **control-plane** option. Therefore, such permitted management traffic will be allowed to come in even if explicitly denied by the to-the-box ACL.

Alternatively, you can use ICMP rules to control ICMP traffic to the device. Use regular extended access rules to control ICMP traffic through the device.

EtherType Rules

This section describes EtherType rules.

Supported EtherTypes and Other Traffic

An EtherType rule controls the following:

- EtherType identified by a 16-bit hexadecimal number, including common types IPX and MPLS unicast or multicast.
- Ethernet V2 frames.
- BPDUs, which are permitted by default. BPDUs are SNAP-encapsulated, and the ASA is designed to specifically handle BPDUs.
- Trunk port (Cisco proprietary) BPDUs. Trunk BPDUs have VLAN information inside the payload, so the ASA modifies the payload with the outgoing VLAN if you allow BPDUs.
- Intermediate System to Intermediate System (IS-IS).

The following types of traffic are not supported:

- 802.3-formatted frames—These frames are not handled by the rule because they use a length field as opposed to a type field.

EtherType Rules for Returning Traffic

Because EtherTypes are connectionless, you need to apply the rule to both interfaces if you want traffic to pass in both directions.

Allowing MPLS

If you allow MPLS, ensure that Label Distribution Protocol and Tag Distribution Protocol TCP connections are established through the ASA by configuring both MPLS routers connected to the ASA to use the IP address on the ASA interface as the router-id for LDP or TDP sessions. (LDP and TDP allow MPLS routers to negotiate the labels (addresses) used to forward packets.)

On Cisco IOS routers, enter the appropriate command for your protocol, LDP or TDP. The *interface* is the interface connected to the ASA.

```
hostname(config)# mpls ldp router-id interface force
```

Or

```
hostname(config)# tag-switching tdp router-id interface force
```

Guidelines for Access Control

IPv6 Guidelines

Supports IPv6. The source and destination addresses can include any mix of IPv4 and IPv6 addresses.

Per-User ACL Guidelines

- The per-user ACL uses the value in the **timeout uauth** command, but it can be overridden by the AAA per-user session timeout value.
- If traffic is denied because of a per-user ACL, syslog message 109025 is logged. If traffic is permitted, no syslog message is generated. The **log** option in the per-user ACL has no effect.

Additional Guidelines and Limitations

- You can reduce the memory required to search access rules by enabling object group search, but this is at the expense rule of lookup performance. When enabled, object group search does not expand network objects, but instead searches access rules for matches based on those group definitions. You can set this option using the **object-group-search access-control** command.
- You can improve system performance and reliability by using the transactional commit model for access groups. See the basic settings chapter in the general operations configuration guide for more information. Use the **asp rule-engine transactional-commit access-group** command.
- In ASDM, rule descriptions are based on the access list remarks that come before the rule in the ACL; for new rules you create in ASDM, any descriptions are also configured as remarks before the related rule. However, the packet tracer in ASDM matches the remark that is configured after the matching rule in the CLI.
- Normally, you cannot reference an object or object group that does not exist in an ACL or object group, or delete one that is currently referenced. You also cannot reference an ACL that does not exist in an **access-group** command (to apply access rules). However, you can change this default behavior so that you can "forward reference" objects or ACLs before you create them. Until you create the objects or ACLs, any rules or access groups that reference them are ignored. To enable forward referencing, use the **forward-reference enable** command.

Configure Access Control

The following topics explain how to configure access control.

- Configure an Access Group, page 4-7
- Configure ICMP Access Rules, page 4-8

Configure an Access Group

Before you can create an access group, create the ACL. See the general operations configuration guide for more information.

To bind an ACL to an interface or to apply it globally, use the following command:

```
access-group access_list {
{in | out} interface interface_name [per-user-override | control-plane] |
global}
```

Example:

```
hostname(config)# access-group outside_access in interface outside
```

For an interface-specific access group:

- Specify the extended or EtherType ACL name. You can configure one **access-group** command per ACL type per interface per direction, and one control plane ACL. The control plane ACL must be an extended ACL.
- The **in** keyword applies the ACL to inbound traffic. The **out** keyword applies the ACL to the outbound traffic.
- Specify the **interface** name.
- The **per-user-override** keyword (for inbound ACLs only) allows dynamic user ACLs that are downloaded for user authorization to override the ACL assigned to the interface. For example, if the interface ACL denies all traffic from 10.0.0.0, but the dynamic ACL permits all traffic from 10.0.0.0, then the dynamic ACL overrides the interface ACL for that user.

  By default, VPN remote access traffic is not matched against interface ACLs. However, if you use the **no sysopt connection permit-vpn** command to turn off this bypass, the behavior depends on whether there is a **vpn-filter** applied in the group policy and whether you set the **per-user-override** option:

  – No **per-user-override**, no **vpn-filter**—Traffic is matched against the interface ACL.

  – No **per-user-override**, **vpn-filter**—Traffic is matched first against the interface ACL, then against the VPN filter.

  – **per-user-override**, **vpn-filter**—Traffic is matched against the VPN filter only.
- The **control-plane** keyword specifies if the rule is for to-the-box traffic.

For a global access group, specify the **global** keyword to apply the extended ACL to the inbound direction of all interfaces.

Examples

The following example shows how to use the **access-group** command:

```
hostname(config)# access-list outside_access permit tcp any host 209.165.201.3 eq 80
hostname(config)# access-group outside_access interface outside
```

The **access-list** command lets any host access the host address using port 80. The **access-group** command specifies that the **access-list** command applies to traffic entering the outside interface.

Configure ICMP Access Rules

By default, you can send ICMP packets to any ASA interface using either IPv4 or IPv6, with these exceptions:

- The ASA does not respond to ICMP echo requests directed to a broadcast address.
- The ASA only responds to ICMP traffic sent to the interface that traffic comes in on; you cannot send ICMP traffic through an interface to a far interface.

To protect the device from attacks, you can use ICMP rules to limit ICMP access to ASA interfaces to particular hosts, networks, or ICMP types. ICMP rules function like access rules, where the rules are ordered, and the first rule that matches a packet defines the action.

If you configure any ICMP rule for an interface, an implicit deny ICMP rule is added to the end of the ICMP rule list, changing the default behavior. Thus, if you want to simply deny a few message types, you must include a permit any rule at the end of the ICMP rule list to allow the remaining message types.

We recommend that you always grant permission for the ICMP unreachable message type (type 3). Denying ICMP unreachable messages disables ICMP path MTU discovery, which can halt IPsec and PPTP traffic. Additionally ICMP packets in IPv6 are used in the IPv6 neighbor discovery process. See RFC 1195 and RFC 1435 for details about path MTU discovery.

Procedure

Step 1 Create rules for ICMP traffic.

```
icmp {permit | deny} {host ip_address | ip_address mask | any}
[icmp_type] interface_name
```

If you do not specify an *icmp_type*, the rule applies to all types. You can enter the number or the name. To control ping, specify echo-reply (0) (ASA-to-host) or echo (8) (host-to-ASA).

For the address, you can apply the rule to **any** address, to a single **host**, or to a network (*ip_address mask*).

Step 2 Create rules for ICMPv6 (IPv6) traffic.

```
ipv6 icmp {permit | deny} {host ipv6_address | ipv6-network/prefix-length | any}
[icmp_type] interface_name
```

If you do not specify an *icmp_type*, the rule applies to all types.

For the address, you can apply the rule to **any** address, to a single **host**, or to a network (*ipv6-network/prefix-length*).

Step 3 (Optional.) Set rate limits on ICMP Unreachable messages so that the ASA will appear on trace route output.

```
icmp unreachable rate-limit rate burst-size size
```

Example

```
hostname(config)# icmp unreachable rate-limit 50 burst-size 1
```

The rate limit can be 1-100, with 1 being the default. The burst size is meaningless, but must be 1-10.

Increasing the rate limit, along with enabling the **set connection decrement-ttl** command in a service policy, is required to allow a traceroute through the ASA that shows the ASA as one of the hops. For example, the following policy decrements the time-to-live (TTL) value for all traffic through the ASA.

```
class-map global-class
  match any
policy-map global_policy
  class global-class
  set connection decrement-ttl
```

Examples

The following example shows how to allow all hosts except the one at 10.1.1.15 to use ICMP to the inside interface:

```
hostname(config)# icmp deny host 10.1.1.15 inside
hostname(config)# icmp permit any inside
```

The following example shows how to allow the host at 10.1.1.15 to use only ping to the inside interface:

```
hostname(config)# icmp permit host 10.1.1.15 inside
```

The following example shows how to deny all ping requests and permit all packet-too-big messages (to support path MTU discovery) at the outside interface:

```
hostname(config)# ipv6 icmp deny any echo-reply outside
hostname(config)# ipv6 icmp permit any packet-too-big outside
```

The following example shows how to permit host 2000:0:0:4::2 or hosts on prefix 2001::/64 to ping the outside interface:

```
hostname(config)# ipv6 icmp permit host 2000:0:0:4::2 echo-reply outside
hostname(config)# ipv6 icmp permit 2001::/64 echo-reply outside
hostname(config)# ipv6 icmp permit any packet-too-big outside
```

Monitoring Access Rules

To monitor network access, enter the following commands:

- **clear access-list** *id* **counters**

  Clear the hit counts for the access list.

- **show access-list** [*name*]

  Displays the access lists, including the line number for each ACE and hit counts. Include an ACL name or you will see all access lists.

- **show running-config access-group**

  Displays the current ACL bound to the interfaces.

Evaluating Syslog Messages for Access Rules

Use a syslog event viewer, such as the one in ASDM, to view messages related to access rules.

If you use default logging, you see syslog message 106023 for explicitly denied flows only. Traffic that matches the “implicit deny” entry that ends the rule list is not logged.

If the ASA is attacked, the number of syslog messages for denied packets can be very large. We recommend that you instead enable logging using syslog message 106100, which provides statistics for each rule (including permit rules) and enables you to limit the number of syslog messages produced. Alternatively, you can disable all logging for a given rule.

When you enable logging for message 106100, if a packet matches an ACE, the ASA creates a flow entry to track the number of packets received within a specific interval. The ASA generates a syslog message at the first hit and at the end of each interval, identifying the total number of hits during the interval and the time stamp for the last hit. At the end of each interval, the ASA resets the hit count to 0. If no packets match the ACE during an interval, the ASA deletes the flow entry. When you configure logging for a rule, you can control the interval and even the severity level of the log message, per rule.

A flow is defined by the source and destination IP addresses, protocols, and ports. Because the source port might differ for a new connection between the same two hosts, you might not see the same flow increment because a new flow was created for the connection.

Permitted packets that belong to established connections do not need to be checked against ACLs; only the initial packet is logged and included in the hit count. For connectionless protocols, such as ICMP, all packets are logged, even if they are permitted, and all denied packets are logged.

See the *syslog messages guide* for detailed information about these messages.

Tip When you enable logging for message 106100, if a packet matches an ACE, the ASA creates a flow entry to track the number of packets received within a specific interval. The ASA has a maximum of 32 K logging flows for ACEs. A large number of flows can exist concurrently at any point of time. To prevent unlimited consumption of memory and CPU resources, the ASA places a limit on the number of concurrent *deny* flows; the limit is placed on deny flows only (not on permit flows) because they can indicate an attack. When the limit is reached, the ASA does not create a new deny flow for logging until the existing flows expire, and issues message 106101. You can control the frequency of this message using the **access-list alert-interval** *secs* command, and the maximum number of deny flows cached using the **access-list deny-flow-max** *number* command.

Configuration Examples for Permitting or Denying Network Access

This section includes typical configuration examples for permitting or denying network access.

The following example adds a network object for inside server 1, performs static NAT for the server, and enables access from the outside for inside server 1.

```
hostname(config)# object network inside-server1
hostname(config)# host 10.1.1.1
hostname(config)# nat (inside,outside) static 209.165.201.12

hostname(config)# access-list outside_access extended permit tcp any object inside-server1
eq www
hostname(config)# access-group outside_access in interface outside
```

The following example allows all hosts to communicate between the **inside** and **hr** networks but only specific hosts to access the outside network:

```
hostname(config)# access-list ANY extended permit ip any any
hostname(config)# access-list OUT extended permit ip host 209.168.200.3 any
hostname(config)# access-list OUT extended permit ip host 209.168.200.4 any

hostname(config)# access-group ANY in interface inside
hostname(config)# access-group ANY in interface hr
hostname(config)# access-group OUT out interface outside
```

For example, the following sample ACL allows common EtherTypes originating on the inside interface:

```
hostname(config)# access-list ETHER ethertype permit ipx
hostname(config)# access-list ETHER ethertype permit mpls-unicast
hostname(config)# access-group ETHER in interface inside
```

The following example allows some EtherTypes through the ASA, but it denies all others:

```
hostname(config)# access-list ETHER ethertype permit 0x1234
hostname(config)# access-list ETHER ethertype permit mpls-unicast
hostname(config)# access-group ETHER in interface inside
hostname(config)# access-group ETHER in interface outside
```

The following example denies traffic with EtherType 0x1256 but allows all others on both interfaces:

```
hostname(config)# access-list nonIP ethertype deny 1256
hostname(config)# access-list nonIP ethertype permit any
hostname(config)# access-group ETHER in interface inside
hostname(config)# access-group ETHER in interface outside
```

The following example uses object groups to permit specific traffic on the inside interface:

```
!
hostname (config)# object-group service myaclog
hostname (config-service)# service-object tcp source range 2000 3000
hostname (config-service)# service-object tcp source range 3000 3010 destinatio$
hostname (config-service)# service-object ipsec
hostname (config-service)# service-object udp destination range 1002 1006
hostname (config-service)# service-object icmp echo

hostname(config)# access-list outsideacl extended permit object-group myaclog interface
inside any
```

History for Access Rules

Feature Name	Platform Releases	Description
Interface access rules	7.0(1)	Controlling network access through the ASA using ACLs. We introduced the following command: **access-group**.
Global access rules	8.3(1)	Global access rules were introduced. We modified the following command: **access-group.**
Support for Identity Firewall	8.4(2)	You can now use identity firewall users and groups for the source and destination. You can use an identity firewall ACL with access rules, AAA rules, and for VPN authentication. We modified the following commands: **access-list extended**.
EtherType ACL support for IS-IS traffic	8.4(5), 9.1(2)	In transparent firewall mode, the ASA can now pass IS-IS traffic using an EtherType ACL. We modified the following command: **access-list ethertype** {**permit** | **deny**} **isis**.
Support for TrustSec	9.0(1)	You can now use TrustSec security groups for the source and destination. You can use an identity firewall ACL with access rules. We modified the following commands: **access-list extended**.

Feature Name	Platform Releases	Description
Unified ACL for IPv4 and IPv6	9.0(1)	ACLs now support IPv4 and IPv6 addresses. You can even specify a mix of IPv4 and IPv6 addresses for the source and destination. The **any** keyword was changed to represent IPv4 and IPv6 traffic. The **any4** and **any6** keywords were added to represent IPv4-only and IPv6-only traffic, respectively. The IPv6-specific ACLs are deprecated. Existing IPv6 ACLs are migrated to extended ACLs. See the release notes for more information about migration. We modified the following commands: **access-list extended**, **access-list webtype**. We removed the following commands: **ipv6 access-list**, **ipv6 access-list webtype**, **ipv6-vpn-filter**
Extended ACL and object enhancement to filter ICMP traffic by ICMP code	9.0(1)	ICMP traffic can now be permitted/denied based on ICMP code. We introduced or modified the following commands: **access-list extended, service-object, service**.
Transactional Commit Model on Access Group Rule Engine	9.1(5)	When enabled, a rule update is applied after the rule compilation is completed; without affecting the rule matching performance. We introduced the following commands: **asp rule-engine transactional-commit**, **show running-config asp rule-engine transactional-commit**, **clear configure asp rule-engine transactional-commit**.
Configuration session for editing ACLs and objects. Forward referencing of objects and ACLs in access rules.	9.3(2)	You can now edit ACLs and objects in an isolated configuration session. You can also forward reference objects and ACLs, that is, configure rules and access groups for objects or ACLs that do not yet exist. We introduced the **clear config-session**, **clear session**, **configure session**, **forward-reference**, and **show config-session** commands.

Identity Firewall

This chapter describes how to configure the ASA for the Identity Firewall.

About the Identity Firewall

In an enterprise, users often need access to one or more server resources. Typically, a firewall is not aware of the users' identities and, therefore, cannot apply security policies based on identity. To configure per-user access policies, you must configure a user authentication proxy, which requires user interaction (a username/password query).

The Identity Firewall in the ASA provides more granular access control based on users' identities. You can configure access rules and security policies based on user names and user group names rather than through source IP addresses. The ASA applies the security policies based on an association of IP addresses to Windows Active Directory login information and reports events based on the mapped usernames instead of network IP addresses.

The Identity Firewall integrates with Microsoft Active Directory in conjunction with an external Active Directory (AD) Agent that provides the actual identity mapping. The ASA uses Windows Active Directory as the source to retrieve the current user identity information for specific IP addresses and allows transparent authentication for Active Directory users.

Identity-based firewall services enhance the existing access control and security policy mechanisms by allowing users or groups to be specified in place of source IP addresses. Identity-based security policies can be interleaved without restriction between traditional IP address-based rules.

The key benefits of the Identity Firewall include:

- Decoupling network topology from security policies
- Simplifying the creation of security policies
- Providing the ability to easily identify user activities on network resources
- Simplifying user activity monitoring

Architecture for Identity Firewall Deployments

The Identity Firewall integrates with Window Active Directory in conjunction with an external Active Directory (AD) Agent that provides the actual identity mapping.

The identity firewall consists of three components:

- ASA
- Microsoft Active Directory

  Although Active Directory is part of the Identity Firewall on the ASA, Active Directory administrators manage it. The reliability and accuracy of the data depends on data in Active Directory.

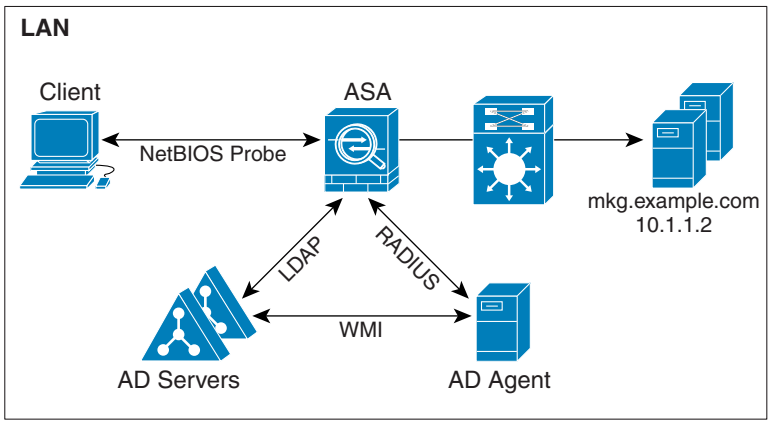
  Supported versions include Windows Server 2003, Windows Server 2008, and Windows Server 2008 R2 servers.
- Active Directory (AD) Agent

  The AD Agent runs on a Windows server. Supported Windows servers include Windows 2003, Windows 2008, and Windows 2008 R2.

Note Windows 2003 R2 is not supported for the AD Agent server.

The following figure show the components of the Identity Firewall. The succeeding table describes the roles of these components and how they communicate with one another.

Figure 5-1 Identity Firewall Components



1	**On the ASA**: Administrators configure local user groups and Identity Firewall policies.	**4**	**Client <-> ASA**: The client logs into the network through Microsoft Active Directory. The AD Server authenticates users and generates user login security logs. Alternatively, the client can log into the network through a cut-through proxy or VPN.
2	**ASA <-> AD Server**: The ASA sends an LDAP query for the Active Directory groups configured on the AD Server. The ASA consolidates local and Active Directory groups and applies access rules and Modular Policy Framework security policies based on user identity.	**5**	**ASA <-> Client**: Based on the policies configured on the ASA, it grants or denies access to the client. If configured, the ASA probes the NetBIOS of the client to pass inactive and no-response users.
3	**ASA <-> AD Agent**: Depending on the Identity Firewall configuration, the ASA downloads the IP-user database or sends a RADIUS request to the AD Agent that asks for the user's IP address. The ASA forwards the new mapped entries that have been learned from web authentication and VPN sessions to the AD Agent.	**6**	**AD Agent <-> AD Server**: The AD Agent maintains a cache of user ID and IP address mapped entries. and notifies the ASA of changes. The AD Agent sends logs to a syslog server.

Features of the Identity Firewall

The Identity Firewall includes the following key features.

Flexibility

- The ASA can retrieve user identity and IP address mapping from the AD Agent by querying the AD Agent for each new IP address or by maintaining a local copy of the entire user identity and IP address database.
- Supports host group, subnet, or IP address for the destination of a user identity policy.

- Supports a fully qualified domain name (FQDN) for the source and destination of a user identity policy.
- Supports the combination of 5-tuple policies with ID-based policies. The identity-based feature works in tandem with the existing 5-tuple solution.
- Supports use with IPS and Application Inspection policies.
- Retrieves user identity information from remote access VPN, AnyConnect VPN, L2TP VPN and cut-through proxy. All retrieved users are populated to all ASAs that are connected to the AD Agent.

Scalability

- Each AD Agent supports 100 ASAs. Multiple ASAs are able to communicate with a single AD Agent to provide scalability in larger network deployments.
- Supports 30 Active Directory servers provided the IP address is unique among all domains.
- Each user identity in a domain can have up to 8 IP addresses.
- Supports up to 64,000 user identity-IP address mapped entries in active policies for the ASA 5500 Series models. This limit controls the maximum number of users who have policies applied. The total number of users are the aggregate of all users configured in all different contexts.
- Supports up to 512 user groups in active ASA policies.
- A single access rule can contain one or more user groups or users.
- Supports multiple domains.

Availability

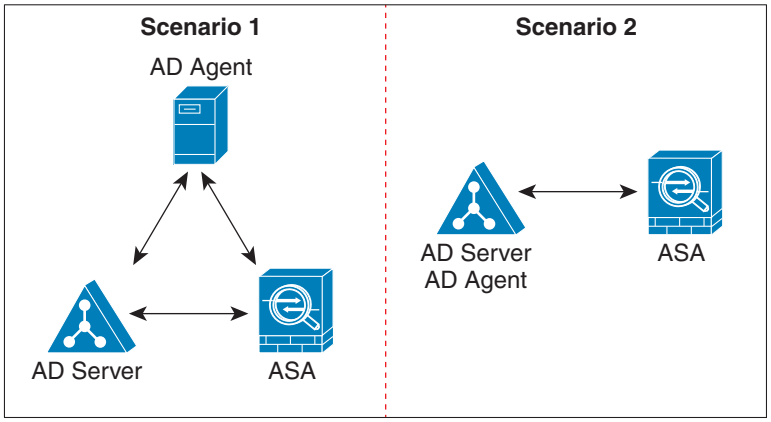
- The ASA retrieves group information from the Active Directory and falls back to web authentication for IP addresses when the AD Agent cannot map a source IP address to a user identity.
- The AD Agent continues to function when any of the Active Directory servers or the ASA are not responding.
- Supports configuring a primary AD Agent and a secondary AD Agent on the ASA. If the primary AD Agent stops responding, the ASA can switch to the secondary AD Agent.
- If the AD Agent is unavailable, the ASA can fall back to existing identity sources such as cut-through proxy and VPN authentication.
- The AD Agent runs a watchdog process that automatically restarts its services when they are down.
- Allows a distributed IP address/user mapping database for use among ASAs.

Deployment Scenarios

You can deploy the components of the Identity Firewall in the following ways, depending on your environmental requirements.

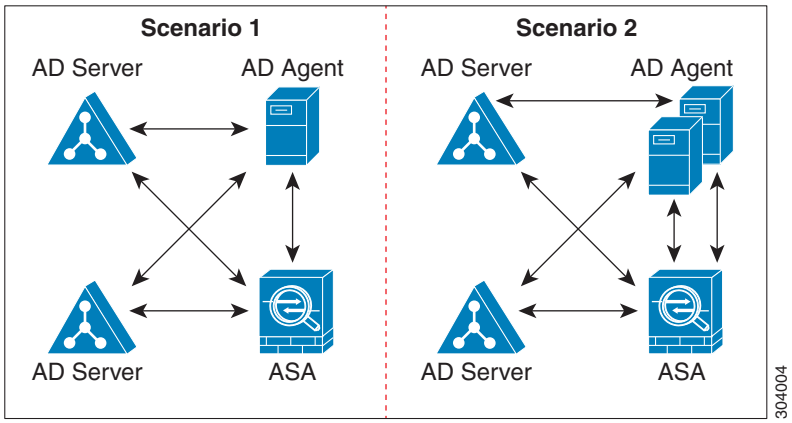
The following figure shows how you can deploy the components of the Identity Firewall to allow for redundancy. Scenario 1 shows a simple installation without component redundancy. Scenario 2 also shows a simple installation without redundancy. However, in this deployment scenario, the Active Directory server and AD Agent are co-located on the same Windows server.

Figure 5-2 Deployment Scenario without Redundancy



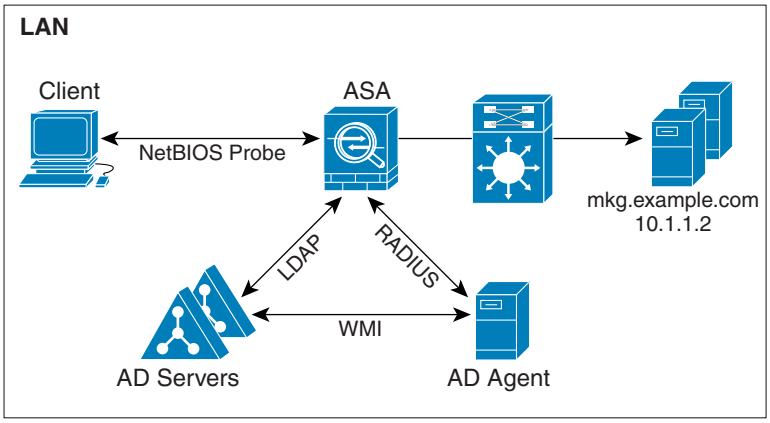
The following figure shows how you can deploy the Identity Firewall components to support redundancy. Scenario 1 shows a deployment with multiple Active Directory servers and a single AD Agent installed on a separate Windows server. Scenario 2 shows a deployment with multiple Active Directory servers and multiple AD Agents installed on separate Windows servers.

Figure 5-3 Deployment Scenario with Redundant Components



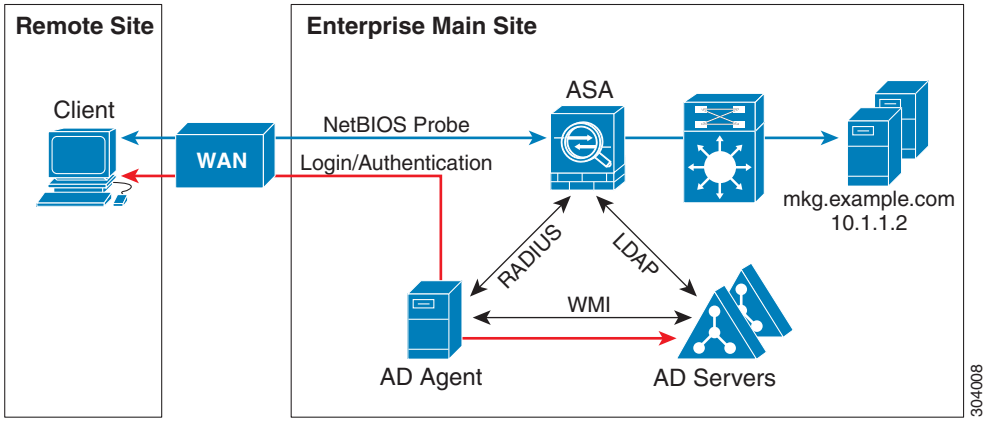
The following figure shows how all Identity Firewall components—Active Directory server, the AD Agent, and the clients—are installed and communicate on the LAN.

Figure 5-4 LAN -based Deployment



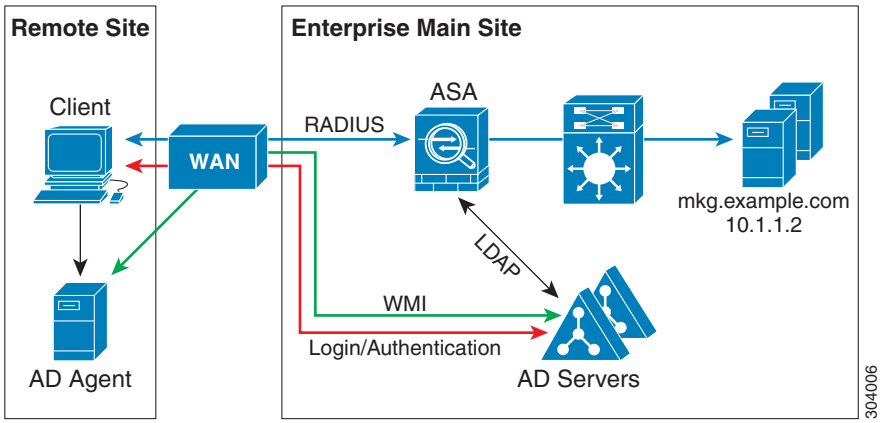
The following figure shows a WAN-based deployment to support a remote site. The Active Directory server and the AD Agent are installed on the main site LAN. The clients are located at a remote site and connect to the Identity Firewall components over a WAN.

Figure 5-5 WAN-based Deployment



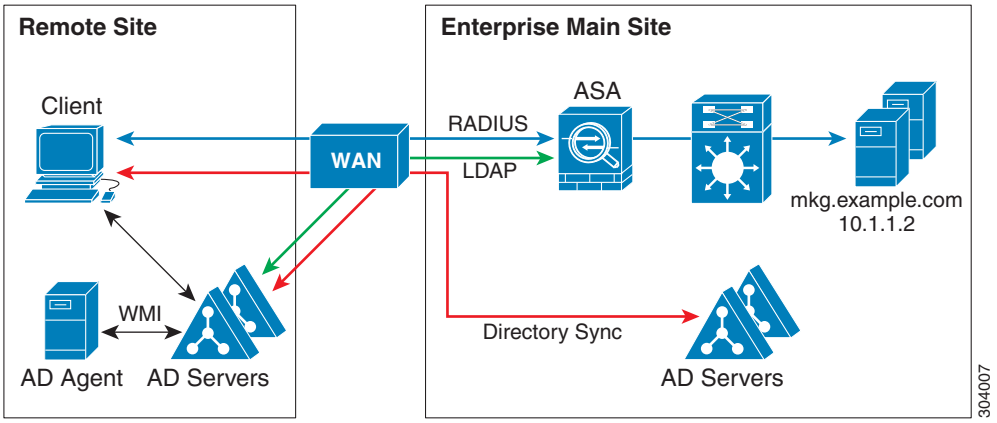
The following figure also shows a WAN-based deployment to support a remote site. The Active Directory server is installed on the main site LAN. However, the AD Agent is installed and accessed by the clients at the remote site. The remote clients connect to the Active Directory servers at the main site over a WAN.

Figure 5-6 WAN-based Deployment with Remote AD Agent



The following figure shows an expanded remote site installation. An AD Agent and Active Directory servers are installed at the remote site. The clients access these components locally when logging into network resources located at the main site. The remote Active Directory server must synchronize its data with the central Active Directory servers located at the main site.

Figure 5-7 WAN-based Deployment with Remote AD Agent and AD Servers



Guidelines for the Identity Firewall

This section describes the guidelines and limitations that you should check before configuring the Identity Firewall.

Failover

- The Identity Firewall supports user identity-IP address mapping and AD Agent status replication from active to standby when Stateful Failover is enabled. However, only user identity-IP address mapping, AD Agent status, and domain status are replicated. User and user group records are not replicated to the standby ASA.
- When failover is configured, the standby ASA must also be configured to connect to the AD Agent directly to retrieve user groups. The standby ASA does not send NetBIOS packets to clients even when the NetBIOS probing options are configured for the Identity Firewall.

- When a client is determined to be inactive by the active ASA, the information is propagated to the standby ASA. User statistics are not propagated to the standby ASA.
- When you have failover configured, you must configure the AD Agent to communicate with both the active and standby ASAs. See the *Installation and Setup Guide for the Active Directory Agent* for the steps to configure the ASA on the AD Agent server.

IPv6

- The AD Agent supports endpoints with IPv6 addresses. It can receive IPv6 addresses in log events, maintain them in its cache, and send them through RADIUS messages. The AAA server must use an IPv4 address.
- NetBIOS over IPv6 is not supported.

Additional Guidelines

- A full URL as a destination address is not supported.
- For NetBIOS probing to function, the network between the ASA, AD Agent, and clients must support UDP-encapsulated NetBIOS traffic.
- MAC address checking by the Identity Firewall does not work when intervening routers are present. Users logged into clients that are behind the same router have the same MAC addresses. With this implementation, all the packets from the same router are able to pass the check, because the ASA is unable to ascertain the actual MAC addresses behind the router.
- The following ASA features do not support using the identity-based object and FQDN in an extended ACL:
  - Route maps
  - Crypto maps
  - WCCP
  - NAT
  - Group policy (except for VPN filters)
  - DAP
- You can use the **user-identity update active-user-database** command to actively initiate a user-IP address download from the AD agent.

  By design, if a previous download session has finished, the ASA does not allow you to issue this command again.

  As a result, if the user-IP database is very large, the previous download session is not finished yet, and you issue another **user-identity update active-user-database** command, the following error message appears:

  ```
  "ERROR: one update active-user-database is already in progress."
  ```

  You need to wait until the previous session is completely finished, then you can issue another **user-identity update active-user-database** command.

  Another example of this behavior occurs because of packet loss from the AD Agent to the ASA.

  When you issue a **user-identity update active-user-database** command, the ASA requests the total number of user-IP mapped entries to be downloaded. Then the AD Agent initiates a UDP connection to the ASA and sends the change of authorization request packet.

If for some reason the packet is lost, there is no way for the ASA to discern this. As a result, the ASA holds the session for 4-5 minutes, during which time this error message continues to appear if you have issued the **user-identity update active-user-database** command.

- When you use the Cisco Context Directory Agent (CDA) in conjunction with the ASA or Cisco Ironport Web Security Appliance (WSA), make sure that you open the following ports:
  - Authentication port for UDP—1645
  - Accounting port for UDP—1646
  - Listening port for UDP—3799

    The listening port is used to send change of authorization requests from the CDA to the ASA or to the WSA.
- If the **user-identity action domain-controller-down** *domain_name* **disable-user-identity-rule** command is configured and the specified domain is down, or if the **user-identity action ad-agent-down disable-user-identity-rule** command is configured and the AD Agent is down, all the logged-in users have the disabled status.
- For domain names, the following characters are not valid: \/:*?"<>|. For naming conventions, see http://support.microsoft.com/kb/909264.
- For usernames, the following characters are not valid: \/[]:;=,+*?"<>|@.
- For user group names, the following characters are not valid: \/[]:;=,+*?"<>|.
- How you configure the Identity Firewall to retrieve user information from the AD Agent affects the amount of memory used by the feature. You specify whether the ASA uses on-demand retrieval or full download retrieval. Choosing on-demand retrieval has the benefit of using less memory, because only users of received packets are queried and stored.

Prerequisites for the Identity Firewall

This section lists the prerequisites for configuring the Identity Firewall.

AD Agent

- The AD Agent must be installed on a Windows server that is accessible to the ASA. Additionally, you must configure the AD Agent to obtain information from the Active Directory servers and to communicate with the ASA.
- Supported Windows servers include Windows 2003, Windows 2008, and Windows 2008 R2.

Note Windows 2003 R2 is not supported for the AD Agent server.

- For the steps to install and configure the AD Agent, see the *Installation and Setup Guide for the Active Directory Agent*.
- Before configuring the AD Agent in the ASA, obtain the secret key value that the AD Agent and the ASA use to communicate. This value must match on both the AD Agent and the ASA.

Microsoft Active Directory

- Microsoft Active Directory must be installed on a Windows server and accessible by the ASA. Supported versions include Windows 2003, 2008, and 2008 R2 servers.

- Before configuring the Active Directory server on the ASA, create a user account in Active Directory for the ASA.
- Additionally, the ASA sends encrypted log-in information to the Active Directory server by using SSL enabled over LDAP. SSL must be enabled on the Active Directory server. See the documentation for Microsoft Active Directory for how to enable SSL for Active Directory.

Note Before running the AD Agent Installer, you must install the patches listed in the *README First for the Cisco Active Directory Agent* on each Microsoft Active Directory server that the AD Agent monitors. These patches are required even when the AD Agent is installed directly on the domain controller server.

Configure the Identity Firewall

To configure the Identity Firewall, perform the following tasks:

Step 1 Configure the Active Directory domain in the ASA.

See Configure the Active Directory Domain, page 5-10.

See also Deployment Scenarios, page 5-4 for the ways in which you can deploy the Active Directory servers to meet your environment requirements.

Step 2 Configure the AD Agent in ASA.

See Configure Active Directory Agents, page 5-13.

See also Deployment Scenarios, page 5-4 for the ways in which you can deploy the AD Agents to meet your environment requirements.

Step 3 Configure Identity Options.

See Configure Identity Options, page 5-14.

Step 4 Configure Identity-based Security Policy. After the AD domain and AD Agent are configured, you can create identity-based object groups and ACLs for use in many features.

See Configure Identity-Based Security Policy, page 5-18.

Configure the Active Directory Domain

Active Directory domain configuration on the ASA is required for the ASA to download Active Directory groups and accept user identities from specific domains when receiving IP-user mapping from the AD Agent.

Before You Begin

- Active Directory server IP address
- Distinguished Name for LDAP base DN
- Distinguished Name and password for the Active Directory user that the Identity Firewall uses to connect to the Active Directory domain controller

To configure the Active Directory domain, perform the following steps:

Procedure

Step 1 Create the AAA server group and configure AAA server parameters for the Active Directory server.

aaa-server *server-tag* **protocol ldap**

Example:

```
hostname(config)# aaa-server adserver protocol ldap
```

Step 2 Configure the AAA server as part of a AAA server group and the AAA server parameters that are host-specific for the Active Directory server.

aaa-server *server-tag* [(*interface-name*)] **host** {*server-ip* | *name*} [*key*] [**timeout** *seconds*]

Example:

```
hostname(config-aaa-server-group)# aaa-server adserver (mgmt) host 172.168.224.6
```

Step 3 Specifies the location in the LDAP hierarchy where the server should begin searching when it receives an authorization request.

ldap-base-dn *string*

Example:

```
hostname(config-aaa-server-host)# ldap-base-dn DC=SAMPLE,DC=com
```

Specifying the **ldap-base-dn** command is optional. If you do not specify this command, the ASA retrieves the defaultNamingContext from the Active Directory and uses it as the base DN.

Step 4 Specify the extent of the search in the LDAP hierarchy that the server should make when it receives an authorization request.

ldap-scope subtree

Example:

```
hostname(config-aaa-server-host)# ldap-scope subtree
```

Step 5 Specify the login password for the LDAP server.

ldap-login-password *string*

Example:

```
hostname(config-aaa-server-host)# ldap-login-password obscurepassword
```

Step 6 Specify the name of the directory object that the system should bind this as.

ldap-login-dn *string*

Example:

```
hostname(config-aaa-server-host)# ldap-login-dn SAMPLE\user1
```

The ASA identifies itself for authenticated binding by attaching a Login DN field to the user authentication request. The Login DN field describes the authentication characteristics of the ASA.

The *string* argument is a case-sensitive string of up to 128 characters that specifies the name of the directory object in the LDAP hierarchy. Spaces are not permitted in the string, but other special characters are allowed.

You can specify the traditional or simplified format.

The typical **ldap-login-dn** command format includes: CN=username,OU=Employees,OU=Sample Users,DC=sample,DC=com.

Step 7 Configure the LDAP server model for the Microsoft Active Directory server.

server-type microsoft

Example:

```
hostname(config-aaa-server-host)# server-type microsoft
```

Step 8 Specify the location of the Active Directory groups configuration in the Active Directory domain controller.

ldap-group-base-dn *string*

Example:

```
hostname(config-aaa-server-host)# ldap-group-base-dn OU=Sample Groups,DC=SAMPLE,DC=com
```

If not specified, the value in the **ldap-group-base-dn** command is used. Specifying this command is optional.

Step 9 Allow the ASA to access the Active Directory domain controller over SSL.

ldap-over-ssl enable

Example:

```
hostname(config-aaa-server-host)# ldap-over-ssl enable
```

To support LDAP over SSL, Active Directory server needs to be configured to have this support.

By default, the Active Directory does not have SSL configured. If SSL is not configured in the Active Directory, you do not need to configure it on the ASA for the Identity Firewall.

Step 10 Specify the server port.

server-port *port-number*

Example:

```
hostname(config-aaa-server-host)# server-port 389
hostname(config-aaa-server-host)# server-port 636
```

By default, if the **ldap-over-ssl** command is not enabled, the default server port is 389; if the **ldap-over-ssl** command is enabled, the default server port is 636.

Step 11 Set the amount of time before LDAP queries time out.

group-search-timeout *seconds*

Example:

```
hostname(config-aaa-server-host)# group-search-timeout 300
```

Configure Active Directory Agents

Configure the primary and secondary AD Agents for the AD Agent Server Group. When the ASA detects that the primary AD Agent is not responding and a secondary agent is specified, the ASA switches to the secondary AD Agent. The Active Directory server for the AD agent uses RADIUS as the communication protocol; therefore, you should specify a key attribute for the shared secret between the ASA and AD Agent.

Before You Begin

- AD agent IP address
- Shared secret between the ASA and AD agent

To configure the AD Agents, perform the following steps:

Procedure

Step 1 Create the AAA server group and configure AAA server parameters for the AD Agent.

aaa-server *server-tag* **protocol radius**

Example:

```
hostname(config)# aaa-server adagent protocol radius
```

Step 2 Enable the AD Agent mode.

ad-agent-mode

Example:

```
hostname(config)# ad-agent-mode
```

Step 3 Configure the AAA server as part of a AAA server group and the AAA server parameters that are host-specific for the AD Agent.

aaa-server *server-tag* [(*interface-name*)] **host** {*server-ip* | *name*} [*key*] [**timeout** *seconds*]

Example:

```
hostname(config-aaa-server-group)# aaa-server adagent (inside) host 192.168.1.101
```

Step 4 Specify the server secret value used to authenticate the ASA to the AD Agent server.

key *key*

Example:

```
hostname(config-aaa-server-host)# key mysecret
```

Step 5 Define the server group of the AD Agent.

user-identity ad-agent aaa-server *aaa_server_group_tag*

Example:

```
hostname(config-aaa-server-hostkey)# user-identity ad-agent aaa-server adagent
```

The first server defined in the *aaa_server_group_tag* argument is the primary AD Agent and the second server defined is the secondary AD Agent. The Identity Firewall supports defining only two AD Agent hosts.

When the ASA detects that the primary AD Agent is down and a secondary agent is specified, it switches to the secondary AD Agent. The AAA server for the AD agent uses RADIUS as the communication protocol, and should specify a key attribute for the shared secret between the ASA and AD Agent.

Step 6 Test the communication between the ASA and the AD Agent server.

test aaa-server ad-agent

Example:

```
hostname(config-aaa-server-host)# test aaa-server ad-agent
```

Configure Identity Options

To configure the Identity Options for the Identity Firewall, perform the following steps:

Procedure

Step 1 Enable the Identity Firewall feature. By default, the Identity Firewall feature is disabled.

user-identity enable

Example:

```
hostname(config)# user-identity enable
```

Step 2 Specify the default domain for the Identity Firewall.

user-identity default-domain *domain_NetBIOS_name*

Example:

```
hostname(config)# user-identity default-domain SAMPLE
```

For the *domain_NetBIOS_name* argument, enter a name of up to 32 characters that consists of [a-z], [A-Z], [0-9], [!@#$%^&()-_=+[]{};,.] except '.' and ' ' at the first character. If the domain name includes a space, enclose the entire name in quotation marks. The domain name is not case sensitive.

The default domain is used for all users and user groups when a domain has not been explicitly configured for those users or groups. When a default domain is not specified, the default domain for users and groups is LOCAL. For multiple context modes, you can set a default domain name for each context, as well as within the system execution space.

Note The default domain name that you specify must match the NetBIOS domain name configured on the Active Directory domain controller. If the domain name does not match, the AD Agent incorrectly associates the user identity-IP address mapped entries with the domain name that you enter when configuring the ASA. To view the NetBIOS domain name, open the Active Directory user event security log in any text editor.

The Identity Firewall uses the LOCAL domain for all locally defined user groups or locally defined users. Users logging in through a web portal (cut-through proxy) are designated as belonging to the Active Directory domain with which they authenticated. Users logging in through a VPN are designated as belonging to the LOCAL domain unless the VPN is authenticated by LDAP with the Active Directory. In this case, the Identity Firewall can associate the users with their Active Directory domain.

Step 3 Associate the LDAP parameters defined for the AAA server for importing user group queries with the domain name.

user-identity domain *domain_nickname* **aaa-server** *aaa_server_group_tag*

Example:

```
hostname(config)# user-identity domain SAMPLE aaa-server ds
```

For the *domain_nickname* argument, enter a name of up to 32 characters consisting of [a-z], [A-Z], [0-9], [!@#$%^&()-_=+[]{};,.] except '.' and ' ' at the first character. If the domain name includes a space, you must enclose that space character in quotation marks. The domain name is not case sensitive.

Step 4 Enable NetBIOS probing.

user-identity logout-probe netbios local-system probe-time minutes *minutes* **retry-interval seconds** *seconds* **retry-count** *times* [**user-not-needed** | **match-any** | **exact-match**]

Example:

```
hostname(config)# user-identity logout-probe netbios local-system probe-time minutes 10
retry-interval seconds 10 retry-count 2 user-not-needed
```

Enabling this option configures how often the ASA probes the user client IP address to determine whether the client is still active. By default, NetBIOS probing is disabled. To minimize the NetBIOS packets, the ASA only sends a NetBIOS probe to a client when the user has been idle for more than the specified number of minutes.

- **Exact-match**—The username of the user assigned to the IP address must be the only one in the NetBIOS response. Otherwise, the user identity of that IP address is considered invalid.
- **User-not-needed**—As long as the ASA received a NetBIOS response from the client, the user identity is considered valid.

The Identity Firewall only performs NetBIOS probing for those users identities that are in the active state and exist in at least one security policy. The ASA does not perform NetBIOS probing for clients where the users logged in through cut-through proxy or by using a VPN.

Step 5 Specify the amount of time before a user is considered idle, meaning the ASA has not received traffic from the user's IP address for the specified amount of time.

user-identity inactive-user-timer minutes *minutes*

Example:

```
hostname(config)# user-identity inactive-user-timer minutes 120
```

When the timer expires, the user's IP address is marked as inactive and removed from the local cached user identity-IP address mapping database, and the ASA no longer notifies the AD Agent about that IP address. Existing traffic is still allowed to pass. When this command is specified, the ASA runs an inactive timer even when the NetBIOS Logout Probe is configured.

By default, the idle timeout is set to 60 minutes. This option does not apply to VPN or cut-through proxy users.

Step 6 Specify the amount of time before the ASA queries the Active Directory server for user group information.

user-identity poll-import-user-group-timer hours *hours*

Example:

```
hostname(config)# user-identity poll-import-user-group-timer hours 1
```

If a user is added to or deleted from an Active Directory group, the ASA received the updated user group after the import group timer ran. By default, the **poll-import-user-group-timer hours** value is 8 hours.

To immediately update user group information, enter the **user-identity update import-user** command.

Step 7 Specify the action when a client does not respond to a NetBIOS probe.

user-identity action netbios-response-fail remove-user-ip

Example:

```
hostname(config)# user-identity action netbios-response-fail remove-user-ip
```

For example, the network connection might be blocked to that client or the client is not active.

When this command is configured, the ASA removes the user identity-IP address mapping for that client.

By default, this command is disabled.

Step 8 Specify the action when the domain is down, because the Active Directory domain controller is not responding.

user-identity action domain-controller-down *domain_nickname* **disable-user-identity-rule**

Example:

```
hostname(config)# user-identity action domain-controller-down SAMPLE
disable-user-identity-rule
```

When the domain is down and the **disable-user-identity-rule** keyword is configured, the ASA disables the user identity-IP address mapping for that domain. Additionally, the status of all user IP addresses in that domain are marked as disabled in the output displayed by the **show user-identity user** command.

By default, this command is disabled.

Step 9 Enable user-not-found tracking. By default, this command is disabled.

user-identity user-not-found enable

Example:

```
hostname(config)# user-identity user-not-found enable
```

Only the last 1024 IP addresses are tracked.

Step 10 Specify the action when the AD Agent is not responding.

user-identity action ad-agent-down disable-user-identity-rule

Example:

```
hostname(config)# user-identity action ad-agent-down disable-user-identity-rule
```

When the AD Agent is down and this command is configured, the ASA disables the user identity rules associated with the users in that domain. Additionally, the status of all user IP addresses in that domain is marked as disabled in the output displayed by the **show user-identity user** command.

By default, this command is disabled.

Step 11 Specify the action when a user's MAC address is found to be inconsistent with the ASA IP address currently mapped to that MAC address.

user-identity action mac-address-mismatch remove-user-ip

Example:

```
hostname(config)# user-identity action mac-address-mismatch remove-user-ip
```

When this command is configured, the ASA removes the user identity-IP address mapping for that client.

By default, the ASA uses the **remove-user-ip** keyword when this command is specified.

Step 12 Define how the ASA retrieves the user identity-IP address mapping information from the AD Agent.

user-identity ad-agent active-user-database {**on-demand** | **full-download**}

Example:

```
hostname(config)# user-identity ad-agent active-user-database full-download
```

By default, the ASA uses the **full-download** option.

- **Full-download**—Specifies that the ASA send a request to the AD Agent to download the entire IP-user mapping table when the ASA starts and then to receive incremental IP-user mapping information when users log in and log out. Full downloads are event driven, meaning that when there are subsequent requests to download the database, just the updates to the user identity-IP address mapping database are sent.
- **On-demand**—Specifies that the ASA retrieve the user mapping information of an IP address from the AD Agent when the ASA receives a packet that requires a new connection, and the user of its source IP address is not in the user-identity database.

When the ASA registers a change request with the AD Agent, the AD Agent sends a new event to the ASA.

Step 13 Define the hello timer between the ASA and the AD Agent.

user-identity ad-agent hello-timer seconds *seconds* **retry-times** *number*

Example:

```
hostname(config)# user-identity ad-agent hello-timer seconds 20 retry-times 3
```

The hello timer between the ASA and the AD Agent defines how frequently the ASA exchanges hello packets. The ASA uses the hello packet to obtain ASA replication status (in-sync or out-of-sync) and domain status (up or down). If the ASA does not receive a response from the AD Agent, it resends a hello packet after the specified interval.

By default, the hello timer is set to 30 seconds and 5 retries.

Step 14 Enable the ASA to keep track of the last event time stamp that it receives for each identifier and to discard any message if the event time stamp is at least 5 minutes older than the ASA's clock, or if its time stamp is earlier than the last event's time stamp.

user-identity ad-agent event-timestamp-check

Example:

```
hostname(config)# user-identity ad-agent event-timestamp-check
```

For a newly booted ASA that does not have knowledge of the last event time stamp, the ASA compares the event time stamp with its own clock. If the event is at least 5 minutes older, the ASA does not accept the message.

We recommend that you configure the ASA, Active Directory, and Active Directory agent to synchronize their clocks among themselves using NTP.

Step 15 Define the server group of the AD Agent.

user-identity ad-agent aaa-server *aaa_server_group_tag*

Example:

```
hostname(config)# user-identity ad-agent aaa-server adagent
```

For the *aaa_server_group_tag* argument, enter the value defined by the **aaa-server** command.

Configure Identity-Based Security Policy

You can incorporate identity-based policy in many ASA features. Any feature that uses extended ACLs (other than those listed as unsupported in the Guidelines for the Identity Firewall, page 5-7) can take advantage of an identity firewall. You can now add user identity arguments to extended ACLs, as well as network-based parameters.

Features that can use identity include the following:

- Access rules—An access rule permits or denies traffic on an interface using network information. With an identity firewall, you can control access based on user identity. See the firewall configuration guide.
- AAA rules—An authentication rule (also known as cut-through proxy) controls network access based on the user. Because this function is very similar to an access rule plus an identity firewall, AAA rules can now be used as a backup method of authentication if a user's AD login expires. For example, for any user without a valid login, you can trigger a AAA rule. To ensure that the AAA rule is only triggered for users that do not have valid logins, you can specify special usernames in the extended ACL used for the access rule and for the AAA rule: None (users without a valid login) and Any (users with a valid login). In the access rule, configure your policy as usual for users and groups, but then include a AAA rule that permits all None users; you must permit these users so they can later trigger a AAA rule. Then, configure a AAA rule that denies Any users (these users are not subject to the AAA rule, and were handled already by the access rule), but permits all None users. For example:

  ```
  access-list 100 ex permit ip user CISCO\xyz any any
  access-list 100 ex deny ip user CISCO\abc any any
  access-list 100 ex permit ip user NONE any any
  access-list 100 ex deny any any
  access-group 100 in interface inside

  access-list 200 ex deny ip user ANY any any
  access-list 200 ex permit user NONE any any
  aaa authenticate match 200 inside user-identity
  ```

  For more information, see the legacy feature guide.
- Cloud Web Security—You can control which users are sent to the Cloud Web Security proxy server. In addition, you can configure policy on the Cloud Web Security ScanCenter that is based on user groups that are included in ASA traffic headers sent to Cloud Web Security. See the firewall configuration guide.
- VPN filter—Although a VPN does not support identity firewall ACLs in general, you can configure the ASA to enforce identity-based access rules on VPN traffic. By default, VPN traffic is not subject to access rules. You can force VPN clients to abide by access rules that use an identity firewall ACL (with the **no sysopt connection permit-vpn** command). You can also use an identity firewall ACL with the VPN filter feature; a VPN filter accomplishes a similar effect by allowing access rules in general.

Related Topics

- Chapter 3, "Access Control Lists."
- Configure Local User Groups, page 2-7

Collect User Statistics

To activate the collection of user statistics by the Modular Policy Framework and match lookup actions for the Identify Firewall, perform the following steps:

Procedure

Step 1 Activate the collection of user statistics by the Modular Policy Framework and matches lookup actions for the Identify Firewall.

user-statistics [**accounting** | **scanning**]

Example:

```
hostname(config)# class-map c-identity-example-1
hostname(config-cmap)# match access-list identity-example-1
hostname(config-cmap)# exit
hostname(config)# policy-map p-identity-example-1
hostname(config-pmap)# class c-identity-example-1
hostname(config-pmap)# user-statistics accounting
hostname(config-pmap)# exit
hostname(config)# service-policy p-identity-example-1 interface outside
```

The **accounting** keyword specifies that the ASA collect the sent packet count, sent drop count, and received packet count. The **scanning** keyword specifies that the ASA collect only the sent drop count.

When you configure a policy map to collect user statistics, the ASA collects detailed statistics for selected users. When you specify the **user-statistics** command without the **accounting** or **scanning** keywords, the ASA collects both accounting and scanning statistics.

Examples for the Identity Firewall

This section provides examples for the Identity Firewall.

AAA Rule and Access Rule Example 1

This example shows a typical cut-through proxy configuration to allow a user to log in through the ASA. In this example, the following conditions apply:

- The ASA IP address is 172.1.1.118.
- The Active Directory domain controller has the IP address 71.1.2.93.
- The end-user client has the IP address 172.1.1.118 and uses HTTPS to log in through a web portal.
- The user is authenticated by the Active Directory domain controller via LDAP.
- The ASA uses the inside interface to connect to the Active Directory domain controller on the corporate network.

```
hostname(config)# access-list AUTH extended permit tcp any 172.1.1.118 255.255.255.255 eq http
hostname(config)# access-list AUTH extended permit tcp any 172.1.1.118 255.255.255.255 eq https
hostname(config)# aaa-server LDAP protocol ldap
hostname(config-aaa-server-group)# aaa-server LDAP (inside) host 171.1.2.93
hostname(config-aaa-server-host)# ldap-base-dn DC=cisco,DC=com
hostname(config-aaa-server-host)# ldap-group-base-dn DC=cisco,DC=com
hostname(config-aaa-server-host)# ldap-scope subtree
hostname(config-aaa-server-host)# ldap-login-dn cn=kao,OU=Employees,OU=Cisco Users,DC=cisco,DC=com
hostname(config-aaa-server-host)# ldap-login-password *****
hostname(config-aaa-server-host)# ldap-over-ssl enable
hostname(config-aaa-server-host)# server-type microsoft
hostname(config-aaa-server-host)# aaa authentication match AUTH inside LDAP
hostname(config)#
hostname(config)# http server enable
hostname(config)# http 0.0.0.0 0.0.0.0 inside
hostname(config)#
hostname(config)# auth-prompt prompt Enter Your Authentication
hostname(config)# auth-prompt accept You are Good
hostname(config)# auth-prompt reject Goodbye
```

AAA Rule and Access Rule Example 2

In this example, the following guidelines apply:

- In **access list** commands, permit user NONE rules should be written before entering the **access-list 100 ex deny any any** command to allow unauthenticated incoming users to trigger AAA cut-through proxy.
- In the **auth access-list** command, permit user NONE rules guarantee only unauthenticated trigger cut-through proxy. Ideally, they should be the last lines.

```
hostname(config)# access-list listenerAuth extended permit tcp any any
hostname(config)# aaa authentication match listenerAuth inside ldap
hostname(config)# aaa authentication listener http inside port 8888
hostname(config)# access-list 100 ex permit ip user SAMPLE\user1 any any
hostname(config)# access-list 100 ex deny ip user SAMPLE\user2 any any
hostname(config)# access-list 100 ex permit ip user NONE any any
hostname(config)# access-list 100 ex deny any any
hostname(config)# access-group 100 in interface inside
hostname(config)# aaa authenticate match 200 inside user-identity
```

VPN Filter Example

Some traffic might need to bypass the Identity Firewall.

The ASA reports users logging in through VPN authentication or a web portal (cut-through proxy) to the AD Agent, which distributes the user information to all registered ASA devices. Specifically, the IP-user mapping of authenticated users is forwarded to all ASA contexts that include the input interface where HTTP/HTTPS packets are received and authenticated. The ASA designates users logging in through a VPN as belonging the LOCAL domain.

There are two different ways to apply identity firewall (IDFW) rules to VPN users:

- Apply VPN-Filter with bypassing access-list check disabled
- Apply VPN-Filter with bypassing access-list check enabled

VPN with IDFW Rule -1 Example

By default, the **sysopt connection permit-vpn** command is enabled and VPN traffic is exempted from an access list check. To apply interface-based ACL rules for VPN traffic, VPN traffic access list bypassing needs to be disabled.

In this example, if the user logs in from the outside interface, the IDFW rules control which network resources are accessible. All VPN users are to be stored under the LOCAL domain. Therefore, it is only meaningful to apply the rules for LOCAL users or object groups that include LOCAL users.

```
! Apply VPN-Filter with bypassing access-list check disabled
no sysopt connection permit-vpn
access-list v1 extended deny ip user LOCAL\idfw any 10.0.0.0 255.255.255.0
access-list v1 extended permit ip user LOCAL\idfw any 20.0.0.0 255.255.255.0
access-group v1 in interface outside
```

VPN with IDFW Rule -2 Example

By default, the **sysopt connection permit-vpn** command is enabled, with VPN traffic access bypassing enabled. A VPN filter can be used to apply the IDFW rules to the VPN traffic. A VPN filter with IDFW rules can be defined in the CLI username and group policy.

In the example, when user idfw logs in, the user can access network resources in the 10.0.00/24 subnet. However, when user user1 logs in, access to network resources in 10.0.00/24 subnet is denied. Note that all VPN users are stored under the LOCAL domain. Therefore, it is only meaningful to apply the rules for LOCAL users or object groups that include LOCAL users.

Note IDFW rules can only be applied to VPN filters under group policy and are not available in all of the other group policy features.

```
! Apply VPN-Filter with bypassing access-list check enabled
sysopt connection permit-vpn
access-list v1 extended permit ip user LOCAL\idfw any 10.0.0.0 255.255.255.0
access-list v2 extended deny ip user LOCAL\user1 any 10.0.0.0 255.255.255.0
username user1 password QkBIIYVi6IFLEsYv encrypted privilege 0 username user1 attributes
    vpn-group-policy group1 vpn-filter value v2
username idfw password eEm2dmjMaopcGozT encrypted
username idfw attributes
    vpn-group-policy testgroup vpn-filter value v1

sysopt connection permit-vpn
access-list v1 extended permit ip user LOCAL\idfw any 10.0.0.0 255.255.255.0 access-list
v1 extended deny ip user LOCAL\user1 any 10.0.0.0 255.255.255.0 group-policy group1
internal
group-policy group1 attributes

    vpn-filter value v1
    vpn-tunnel-protocol ikev1 l2tp-ipsec ssl-client ssl-clientless
```

Monitoring the Identity Firewall

See the following commands for monitoring the Identity Firewall status:

- **show user-identity ad-agent**

This command shows the status of the AD Agent and the domains.

- **show user-identity ad-agent statistics**

  This command shows the statistics for the AD Agent.

- **show user-identity memory**

  This command shows the memory usage of various modules in the Identity Firewall.

- **show user-identity user all list**

  This command shows information about all users contained in the IP-user mapping database used by the Identity Firewall.

- **show user-identity user active user** *domain\user-name* **list detail**

  This command shows additional information about an active user.

- **show user-identity group**

  This command shows the list of user groups configured for the Identity Firewall.

History for the Identity Firewall

Table 5-1 History for the Identity Firewall

Feature Name	Releases	Description
Identity Firewall	8.4(2)	The Identity Firewall feature was introduced. We introduced or modified the following commands: **user-identity enable, user-identity default-domain, user-identity domain, user-identity logout-probe, user-identity inactive-user-timer, user-identity poll-import-user-group-timer, user-identity action netbios-response-fail, user-identity user-not-found, user-identity action ad-agent-down, user-identity action mac-address-mismatch, user-identity action domain-controller-down, user-identity ad-agent active-user-database, user-identity ad-agent hello-timer, user-identity ad-agent aaa-server, user-identity update import-user**, **user-identity static user**, **dns domain-lookup**, **dns poll-timer**, **dns expire-entry-timer**, **object-group user, show user-identity**, **show dns, clear configure user-identity**, **clear dns, debug user-identity**.

ASA and Cisco TrustSec

This chapter describes how to implement Cisco TrustSec for the ASA.

About Cisco TrustSec

Traditionally, security features such as firewalls performed access control based on predefined IP addresses, subnets, and protocols. However, with enterprises transitioning to borderless networks, both the technology used to connect people and organizations and the security requirements for protecting data and networks have evolved significantly. Endpoints are becoming increasingly nomadic and users often employ a variety of endpoints (for example, laptop versus desktop, smart phone, or tablet), which means that a combination of user attributes plus endpoint attributes provide the key characteristics (in addition to existing 6-tuple based rules), that enforcement devices such as switches and routers with firewall features or dedicated firewalls can reliably use for making access control decisions.

As a result, the availability and propagation of endpoint attributes or client identity attributes have become increasingly important requirements to enable security across the customers' networks, at the access, distribution, and core layers of the network, and in the data center.

Cisco TrustSec provides access control that builds upon an existing identity-aware infrastructure to ensure data confidentiality between network devices and integrate security access services on one platform. In the Cisco TrustSec feature, enforcement devices use a combination of user attributes and endpoint attributes to make role-based and identity-based access control decisions. The availability and propagation of this information enables security across networks at the access, distribution, and core layers of the network.

Implementing Cisco TrustSec into your environment has the following advantages:

- Provides a growing mobile and complex workforce with appropriate and more secure access from any device
- Lowers security risks by providing comprehensive visibility of who and what is connecting to the wired or wireless network

- Offers exceptional control over activity of network users accessing physical or cloud-based IT resources
- Reduces total cost of ownership through centralized, highly secure access policy management and scalable enforcement mechanisms
- For more information, see the following URLs:

Reference	Description
http://www.cisco.com/c/en/us/solutions/enterprise-networks/trustsec/index.html	Describes the Cisco TrustSec system and architecture for the enterprise.
http://www.cisco.com/c/en/us/solutions/enterprise/design-zone-security/landing_DesignZone_TrustSec.html	Provides instructions for deploying the Cisco TrustSec solution in the enterprise, including links to component design guides.
http://www.cisco.com/c/en/us/solutions/collateral/enterprise-networks/trustsec/solution_overview_c22-591771.pdf	Provides an overview of the Cisco TrustSec solution when used with the ASA, switches, wireless LAN (WLAN) controllers, and routers.
http://www.cisco.com/c/en/us/solutions/enterprise-networks/trustsec/trustsec_matrix.html	Provides the Cisco TrustSec Platform Support Matrix, which lists the Cisco products that support the Cisco TrustSec solution.

About SGT and SXP Support in Cisco TrustSec

In the Cisco TrustSec feature, security group access transforms a topology-aware network into a role-based network, which enables end-to-end policies enforced on the basis of role-based access control (RBAC). Device and user credentials acquired during authentication are used to classify packets by security groups. Every packet entering the Cisco TrustSec cloud is tagged with a security group tag (SGT). The tagging helps trusted intermediaries identify the source identity of the packet and enforce security policies along the data path. An SGT can indicate a privilege level across the domain when the SGT is used to define a security group ACL.

An SGT is assigned to a device through IEEE 802.1X authentication, web authentication, or MAC authentication bypass (MAB), which occurs with a RADIUS vendor-specific attribute. An SGT can be assigned statically to a particular IP address or to a switch interface. An SGT is passed along dynamically to a switch or access point after successful authentication.

The Security-group eXchange Protocol (SXP) is a protocol developed for Cisco TrustSec to propagate the IP-to-SGT mapping database across network devices that do not have SGT-capable hardware support to hardware that supports SGTs and security group ACLs. SXP, a control plane protocol, passes IP-SGT mapping from authentication points (such as legacy access layer switches) to upstream devices in the network.

The SXP connections are point-to-point and use TCP as the underlying transport protocol. SXP uses the well-known TCP port number 64999 to initiate a connection. Additionally, an SXP connection is uniquely identified by the source and destination IP addresses.

Roles in the Cisco TrustSec Feature

To provide identity and policy-based access enforcement, the Cisco TrustSec feature includes the following roles:

- Access Requester (AR)—Access requesters are endpoint devices that request access to protected resources in the network. They are primary subjects of the architecture and their access privilege depends on their Identity credentials.

  Access requesters include endpoint devices such PCs, laptops, mobile phones, printers, cameras, and MACsec-capable IP phones.

- Policy Decision Point (PDP)—A policy decision point is responsible for making access control decisions. The PDP provides features such as 802.1x, MAB, and web authentication. The PDP supports authorization and enforcement through VLAN, DACL, and security group access (SGACL/SXP/SGT).

  In the Cisco TrustSec feature, the Cisco Identity Services Engine (ISE) acts as the PDP. The Cisco ISE provides identity and access control policy functionality.

- Policy Information Point (PIP)—A policy information point is a source that provides external information (for example, reputation, location, and LDAP attributes) to policy decision points.

  Policy information points include devices such as Session Directory, Sensor IPS, and Communication Manager.

- Policy Administration Point (PAP)—A policy administration point defines and inserts policies into the authorization system. The PAP acts as an identity repository by providing Cisco TrustSec tag-to-user identity mapping and Cisco TrustSec tag-to-server resource mapping.

  In the Cisco TrustSec feature, the Cisco Secure Access Control System (a policy server with integrated 802.1x and SGT support) acts as the PAP.

- Policy Enforcement Point (PEP)—A policy enforcement point is the entity that carries out the decisions (policy rules and actions) made by the PDP for each AR. PEP devices learn identity information through the primary communication path that exists across networks. PEP devices learn the identity attributes of each AR from many sources, such as endpoint agents, authorization servers, peer enforcement devices, and network flows. In turn, PEP devices use SXP to propagate IP-SGT mapping to mutually trusted peer devices across the network.

  Policy enforcement points include network devices such as Catalyst switches, routers, firewalls (specifically the ASA), servers, VPN devices, and SAN devices.

The Cisco ASA serves the PEP role in the identity architecture. Using SXP, the ASA learns identity information directly from authentication points and uses it to enforce identity-based policies.

Security Group Policy Enforcement

Security policy enforcement is based on security group name. An endpoint device attempts to access a resource in the data center. Compared to traditional IP-based policies configured on firewalls, identity-based policies are configured based on user and device identities. For example, mktg-contractor is allowed to access mktg-servers; mktg-corp-users are allowed to access mktg-server and corp-servers.

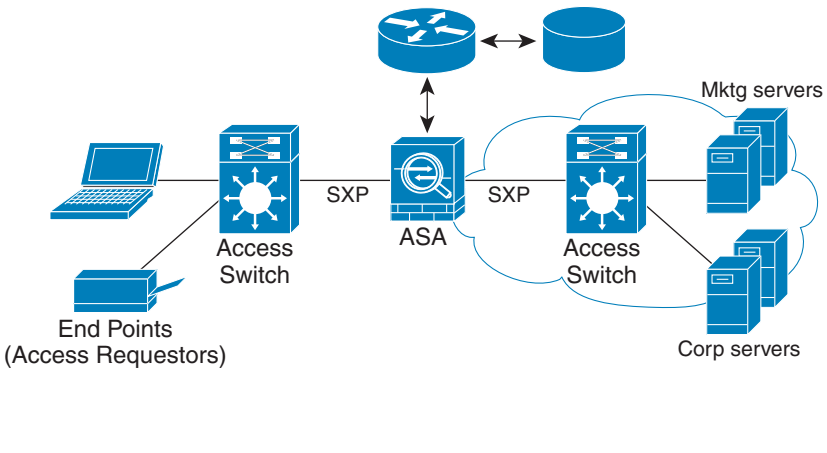
The benefits of this type of deployment include the following:

- User group and resource are defined and enforced using single object (SGT) simplified policy management.

- User identity and resource identity are retained throughout the Cisco TrustSec-capable switch infrastructure.

The following figure shows a deployment for security group name-based policy enforcement.

Figure 6-1 Security Group Name-Based Policy Enforcement Deployment



Implementing Cisco TrustSec allows you to configure security policies that support server segmentation and includes the following features:

- A pool of servers can be assigned an SGT for simplified policy management.
- The SGT information is retained within the infrastructure of Cisco TrustSec-capable switches.
- The ASA can use the IP-SGT mapping for policy enforcement across the Cisco TrustSec domain.
- Deployment simplification is possible because 802.1x authorization for servers is mandatory.

How the ASA Enforces Security Group-Based Policies

Note User-based security policies and security-group based policies can coexist on the ASA. Any combination of network, user-based, and security-group based attributes can be configured in a security policy.

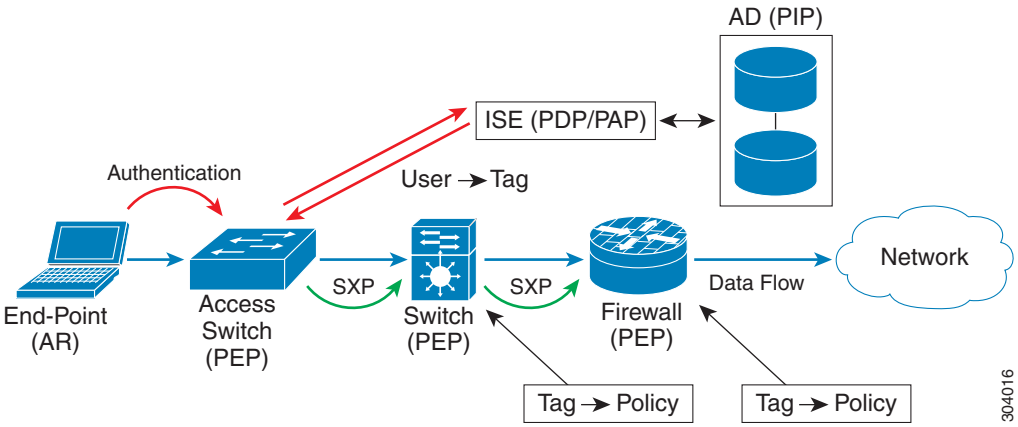
To configure the ASA to function with Cisco TrustSec, you must import a Protected Access Credential (PAC) file from the ISE.

Importing the PAC file to the ASA establishes a secure communication channel with the ISE. After the channel is established, the ASA initiates a PAC secure RADIUS transaction with the ISE and downloads Cisco TrustSec environment data (that is, the security group table). The security group table maps SGTs to security group names. Security group names are created on the ISE and provide user-friendly names for security groups.

The first time that the ASA downloads the security group table, it walks through all entries in the table and resolves all the security group names included in security policies that have been configured on it; then the ASA activates those security policies locally. If the ASA cannot resolve a security group name, it generates a syslog message for the unknown security group name.

The following figure shows how a security policy is enforced in Cisco TrustSec.

Figure 6-2 Security Policy Enforcement



1. An endpoint device connects to an access layer device directly or via remote access and authenticates with Cisco TrustSec.
2. The access layer device authenticates the endpoint device with the ISE by using authentication methods such as 802.1X or web authentication. The endpoint device passes role and group membership information to classify the device into the appropriate security group.
3. The access layer device uses SXP to propagate the IP-SGT mapping to the upstream devices.
4. The ASA receives the packet and looks up the SGTs for the source and destination IP addresses using the IP-SGT mapping passed by SXP.

   If the mapping is new, the ASA records it in its local IP-SGT Manager database. The IP-SGT Manager database, which runs in the control plane, tracks IP-SGT mapping for each IPv4 or IPv6 address. The database records the source from which the mapping was learned. The peer IP address of the SXP connection is used as the source of the mapping. Multiple sources can exist for each IP-SGT mapped entry.

   If the ASA is configured as a Speaker, the ASA transmits all IP-SGT mapping entries to its SXP peers.
5. If a security policy is configured on the ASA with that SGT or security group name, the ASA enforces the policy. (You can create security policies on the ASA that include SGTs or security group names. To enforce policies based on security group names, the ASA needs the security group table to map security group names to SGTs.)

   If the ASA cannot find a security group name in the security group table and it is included in a security policy, the ASA considers the security group name to be unknown and generates a syslog message. After the ASA refreshes the security group table from the ISE and learns the security group name, the ASA generates a syslog message indicating that the security group name is known.

Effects of Changes to Security Groups on the ISE

The ASA periodically refreshes the security group table by downloading an updated table from the ISE. Security groups can change on the ISE between downloads. These changes are not reflected on the ASA until it refreshes the security group table.

Tip We recommend that you schedule policy configuration changes on the ISE during a maintenance window, then manually refresh the security group table on the ASA to make sure the security group changes have been incorporated.

Handling policy configuration changes in this way maximizes the chances of security group name resolution and immediate activation of security policies.

The security group table is automatically refreshed when the environment data timer expires. You can also trigger a security group table refresh on demand.

If a security group changes on the ISE, the following events occur when the ASA refreshes the security group table:

- Only security group policies that have been configured using security group names need to be resolved with the security group table. Policies that include security group tags are always active.
- When the security group table is available for the first time, all policies with security group names are walked through, security group names are resolved, and policies are activated. All policies with tags are walked through, and syslogs are generated for unknown tags.
- If the security group table has expired, policies continue to be enforced according to the most recently downloaded security group table until you clear it, or a new table becomes available.
- When a resolved security group name becomes unknown on the ASA, it deactivates the security policy; however, the security policy persists in the ASA running configuration.
- If an existing security group is deleted on the PAP, a previously known security group tag can become unknown, but no change in policy status occurs on the ASA. A previously known security group name can become unresolved, and the policy is then inactivated. If the security group name is reused, the policy is recompiled using the new tag.
- If a new security group is added on the PAP, a previously unknown security group tag can become known, a syslog message is generated, but no change in policy status occurs. A previously unknown security group name can become resolved, and associated policies are then activated.
- If a tag has been renamed on the PAP, policies that were configured using tags display the new name, and no change in policy status occurs. Policies that were configured with security group names are recompiled using the new tag value.

Speaker and Listener Roles on the ASA

The ASA supports SXP to send and receive IP-SGT mapping entries to and from other network devices. Using SXP allows security devices and firewalls to learn identity information from access switches without the need for hardware upgrades or changes. SXP can also be used to pass IP-SGT mapping entries from upstream devices (such as data center devices) back to downstream devices. The ASA can receive information from both upstream and downstream directions.

When configuring an SXP connection on the ASA to an SXP peer, you must designate the ASA as a Speaker or a Listener for that connection so that it can exchange Identity information:

- Speaker mode—Configures the ASA so that it can forward all active IP-SGT mapping entries collected on the ASA to upstream devices for policy enforcement.
- Listener mode—Configures the ASA so that it can receive IP-SGT mapping entries from downstream devices (SGT-capable switches) and use that information to create policy definitions.

If one end of an SXP connection is configured as a Speaker, then the other end must be configured as a Listener, and vice versa. If both devices on each end of an SXP connection are configured with the same role (either both as Speakers or both as Listeners), the SXP connection fails and the ASA generates a syslog message.

Multiple SXP connections can learn IP-SGT mapping entries that have been downloaded from the IP-SGT mapping database. After an SXP connection to an SXP peer is established on the ASA, the Listener downloads the entire IP-SGT mapping database from the Speaker. All changes that occur after this are sent only when a new device appears on the network. As a result, the rate of SXP information flow is proportional to the rate at which end hosts authenticate to the network.

IP-SGT mapping entries that have been learned through SXP connections are maintained in the SXP IP-SGT mapping database. The same mapping entries may be learned through different SXP connections. The mapping database maintains one copy for each mapping entry learned. Multiple mapping entries of the same IP-SGT mapping value are identified by the peer IP address of the connection from which the mapping was learned. SXP requests that the IP-SGT Manager add a mapping entry when a new mapping is learned the first time and remove a mapping entry when the last copy in the SXP database is removed.

Whenever an SXP connection is configured as a Speaker, SXP requests that the IP-SGT Manager forward all the mapping entries collected on the device to the peer. When a new mapping is learned locally, the IP-SGT Manager requests that SXP forward it through connections that are configured as Speakers.

Configuring the ASA to be both a Speaker and a Listener for an SXP connection can cause SXP looping, which means that SXP data can be received by an SXP peer that originally transmitted it.

SXP Chattiness

The rate of SXP information flow is proportional to the rate at which end hosts authenticate into the network. After an SXP peering is established, the listener device downloads the entire IP-SGT database from the speaker device. After that, all changes are sent incrementally only when a new device appears on the network or leaves the network. Also, note that only access devices that are attached to the new device initiate this incremental update to the upstream device.

In other words, SXP protocol is no chattier than the authentication rate, which is limited to the capability of the authentication server. Therefore, SXP chattiness is not a major concern.

SXP Timers

- Retry Open Timer—The retry open timer is triggered if one SXP connection on the device is not up. After the retry open timer expires, the device goes through the entire connection database and if any connection is in the off or “pending on” state, the retry open timer restarts. The default timer value is 120 seconds. A zero value means the retry timer does not start. The retry open timer continues until all the SXP connections are set up, or the retry open timer has been configured to be 0.
- Delete Hold-Down Timer—The connection-specific delete hold-down timer is triggered when a connection on the Listener is torn down. The mapping entries that have been learned are not deleted immediately, but are held until the delete hold-down timer expires. The mapping entries are deleted after this timer expires. The delete hold-down timer value is set to 120 seconds and is not configurable.
- Reconciliation Timer—If an SXP connection is brought up within the delete hold-down timer period, a bulk update is performed on this connection. This means that the most recent mapping entries are learned and are associated with a new connection instantiation identifier. A periodic,

connection-specific reconciliation timer starts in the background. When this reconciliation timer expires, it scans the entire SXP mapping database and identifies all mapping entries that have not been learned in the current connection session (that is, mapping entries with an unmatched connection instantiation identifier), and marks them for deletion. These entries are deleted in the subsequent reconciliation review. The default reconciliation timer value is 120 seconds. A zero value is not allowed on the ASA to prevent obsolete entries from staying for an unspecified length of time and causing unexpected results in policy enforcement.

- HA Reconciliation Timer—When HA is enabled, the SXP mapping database of the active and standby units are in sync. The new active unit tries to establish new SXP connections to all its peers and acquires the latest mapping entries. An HA reconciliation timer provides a way of identifying and removing old mapping entries. It starts after a failover occurs, which gives the ASA time to acquire the latest mapping entries. After the HA reconciliation timer expires, the ASA scans the entire SXP mapping database and identifies all the mapping entries have not been learned in the current connection session. Mapping entries with unmatched instantiation identifiers are marked for deletion. This reconciliation mechanism is the same as that of the reconciliation timer. The time value is the same as the reconciliation timer and is configurable.

  After an SXP peer terminates its SXP connection, the ASA starts a delete hold-down timer. Only SXP peers designated as Listeners can terminate a connection. If an SXP peer connects while the delete hold-down timer is running, the ASA starts the reconciliation timer; then the ASA updates the IP-SGT mapping database to learn the most recent mapping.

IP-SGT Manager Database

The IP-SGT Manager database does not synchronize any entries from the active unit to the standby unit. Each source from which the IP-SGT Manager database receives IP-SGT mapping entries synchronizes its database from the active unit to the standby unit, then provides the final IP-SGT mapping to the IP-SGT Manager on the standby unit.

For Version 9.0(1), the IP-SGT Manager database receives IP-SGT mapping updates from the SXP source only.

Features of the ASA-Cisco TrustSec Integration

Cisco TrustSec provides the following capabilities:

Flexibility

- The ASA can be configured as an SXP Speaker or Listener, or both.
- The ASA supports SXP for IPv6 and IPv6-capable network devices.
- SXP can change mapping entries for IPv4 and IPv6 addresses.
- SXP endpoints support IPv4 and IPv6 addresses.
- The ASA supports SXP Version 2 only.
- The ASA negotiates SXP versions with different SXP-capable network devices. SXP version negotiation eliminates the need for static configuration of versions.
- You can configure the ASA to refresh the security group table when the SXP reconcile timer expires and you can download the security group table on demand. When the security group table on the ASA is updated from the ISE, changes are reflected in the appropriate security policies.

- The ASA supports security policies based on security group names in the source or destination fields, or both. You can configure security policies on the ASA based on combinations of security groups, IP address, Active Directory group/user name, and FQDN.

Availability

- You can configure security group-based policies on the ASA in both the Active/Active and Active/Standby configurations.
- The ASA can communicate with the ISE configured for high availability (HA).
- You can configure multiple ISE servers on the ASA and if the first server is unreachable, it continues to the next server, and so on. However, if the server list is downloaded as part of the Cisco TrustSec environment data, it is ignored.
- If the PAC file downloaded from the ISE expires on the ASA and it cannot download an updated security group table, the ASA continues to enforce security policies based on the last downloaded security group table until the ASA downloads an updated table.

Clustering

- For Layer 2 networks, all units share the same IP address. When you change the interface address, the changed configuration is sent to all other units. When the IP address is updated from the interface of a particular unit, a notification is sent to update the IP-SGT local database on this unit.
- For Layer 3 networks, a pool of addresses is configured for each interface on the master unit, and this configuration is synchronized to the slave units. On the master unit, a notification of the IP addresses that have been assigned to the interface is sent, and the IP-SGT local database is updated. The IP-SGT local database on each slave unit can be updated with the IP address information for the master unit by using the address pool configuration that has been synchronized to it, where the first address in the pool for each interface always belongs to the master unit.

  When a slave unit boots, it notifies the master unit. Then the master unit goes through the address pool on each interface and computes the IP address for the new slave unit that sent it the notification, and updates the IP-SGT local database on the master unit. The master unit also notifies the other slave units about the new slave unit. As part of this notification processing, each slave unit computes the IP address for the new slave unit and adds this entry to the IP-SGT local database on each slave unit. All the slave units have the address pool configuration to determine the IP address value. For each interface, the value is determined as follows:

  Master IP + (M-N), where:

  M—Maximum number of units (up to 8 are allowed)

  N—Slave unit number that sent the notification

  When the IP address pool changes on any interface, the IP addresses for all the slave units and the master unit need to be recalculated and updated in the IP-SGT local database on the master unit, as well as on every other slave unit. The old IP address needs to be deleted, and the new IP address needs to be added.

  When this changed address pool configuration is synchronized to the slave unit, as a part of configuration change processing, each slave unit recomputes the IP address for the master unit and for every other slave unit whose IP address has changed, then removes the entry for the old IP address and adds the new IP address.

Scalability

The following table shows the number of IP-SGT mapping entries that the ASA supports.

Table 6-1 Capacity Numbers for IP-SGT Mapping Entries

ASA Model	Number of IP-SGT Mapping Entries
5585-X with SSP-10	18,750
5585-X with SSP-20	25,000
5585-X with SSP-40	50,000
5585-X with SSP-60	100,000

The following table shows the number of SXP connections that the ASA supports.

Table 6-2 SXP Connections

ASA Model	Number of SXP TCP Connections
5585-X with SSP-10	150
5585-X with SSP-20	250
5585-X with SSP-40	500
5585-X with SSP-60	1000

Register the ASA with the ISE

The ASA must be configured as a recognized Cisco TrustSec network device in the ISE before the ASA can successfully import a PAC file. To register the ASA with the ISE, perform the following steps:

1. Log into the ISE.
2. Choose **Administration > Network Devices > Network Devices**.
3. Click **Add**.
4. Enter the IP address of the ASA.
5. When the ISE is being used for user authentication, enter a shared secret in the Authentication Settings area.

   When you configure the AAA sever on the ASA, provide the shared secret that you create here on the ISE. The AAA server on the ASA uses this shared secret to communicate with the ISE.
6. Specify a device name, device ID, password, and a download interval for the ASA. See the ISE documentation for how to perform these tasks.

Create a Security Group on the ISE

When configuring the ASA to communicate with the ISE, you specify a AAA server. When configuring the AAA server on the ASA, you must specify a server group. The security group must be configured to use the RADIUS protocol. To create a security group on the ISE, perform the following steps:

1. Log into the ISE.

2. Choose **Policy > Policy Elements > Results > Security Group Access > Security Group**.
3. Add a security group for the ASA. (Security groups are global and not ASA specific.)

   The ISE creates an entry under Security Groups with a tag.
4. In the Security Group Access area, configure device ID credentials and a password for the ASA.

Generate the PAC File

To generate the PAC file, perform the following steps:

1. Log into the ISE.
2. Choose **Administration > Network Resources > Network Devices**.
3. From the list of devices, choose the ASA.
4. Under the Security Group Access (SGA), click **Generate PAC**.
5. To encrypt the PAC file, enter a password.

   The password (or encryption key) that you enter to encrypt the PAC file is independent of the password that was configured on the ISE as part of the device credentials.

The ISE generates the PAC file. The ASA can import the PAC file from flash or from a remote server via TFTP, FTP, HTTP, HTTPS, or SMB. (The PAC file does not have to reside on the ASA flash before you can import it.)

Note The PAC file includes a shared key that allows the ASA and ISE to secure the RADIUS transactions that occur between them. For this reason, make sure that you store it securely on the ASA.

Guidelines for Cisco TrustSec

This section includes the guidelines and limitations that you should review before configuring Cisco TrustSec.

Failover

- Supports a list of servers via configuration. If the first server is unreachable, the ASA tries to contact the second server in the list, and so on. However, the server list downloaded as part of the Cisco TrustSec environment data is ignored.
- When the ASA is part of a failover configuration, you must import the PAC file to the primary ASA device.
- When the ASA is part of a failover configuration, you must refresh the environment data on the primary ASA device.

Clustering

- When the ASA is part of a clustering configuration, you must import the PAC file to the master unit.
- When the ASA is part of a clustering configuration, you must refresh the environment data on the master unit.

IPv6

The ASA supports SXP for IPv6 and IPv6-capable network devices. The AAA server must use an IPv4 address.

Layer 2 SGT Imposition

- Supported only on physical interfaces, VLAN interfaces, port channel interfaces, and redundant interfaces.
- Not supported on logical interfaces or virtual interfaces, such as BVI.
- Does not support link encryption using SAP negotiation and MACsec.
- Not supported on failover links.
- Not supported on cluster control links.
- The ASA does not reclassify existing flows if the SGT is changed. Any policy decisions that were made based on the previous SGT remain in force for the life of the flow. However, the ASA can immediately reflect SGT changes on egress packets, even if the packets belong to a flow whose classification was based on a previous SGT.
- The hardware architecture of the ASA 5585-X is designed to load balance regular packets in an optimal way, but this is not the case for inline tagged packets with Layer 2 Security Group Tagging Imposition. Significant performance degradation on the ASA 5585-X may occur when it processes incoming inline tagged packets. This issue does not occur with inline tagged packets on other ASA platforms, as well as with untagged packets on the ASA 5585-X. One workaround is to offload access policies so that minimal inline tagged packets go to the ASA 5585-X, which allows the switches to handle tagged policy enforcement. Another workaround is to use SXP so that the ASA 5585-X can map the IP address to the security group tag without the need to receive tagged packets.
- The ASASM does not support Layer 2 Security Group Tagging Imposition.

Additional Guidelines

- Cisco TrustSec supports the Smart Call Home feature in single context and multi-context mode, but not in the system context.
- The ASA can only be configured to interoperate in a single Cisco TrustSec domain.
- The ASA does not support static configuration of SGT-name mapping on the device.
- NAT is not supported in SXP messages.
- SXP conveys IP-SGT mapping to enforcement points in the network. If an access layer switch belongs to a different NAT domain than the enforcing point, the IP-SGT map that it uploads is invalid, and an IP-SGT mapping database lookup on the enforcement device does not yield valid results. As a result, the ASA cannot apply security group-aware security policy on the enforcement device.
- You can configure a default password for the ASA to use for SXP connections, or you can choose not to use a password; however, connection-specific passwords are not supported for SXP peers. The configured default SXP password should be consistent across the deployment network. If you configure a connection-specific password, connections may fail and a warning message appears. If you configure the connection with the default password, but it is not configured, the result is the same as when you have configured the connection with no password.
- SXP connection loops can form when a device has bidirectional connections to a peer or is part of a unidirectionally connected chain of devices. (The ASA can learn IP-SGT mapping for resources from the access layer in the data center. The ASA might need to propagate these tags to downstream

devices.) SXP connection loops can cause unexpected behavior of SXP message transport. In cases where the ASA is configured to be a Speaker and Listener, an SXP connection loop can occur, causing SXP data to be received by the peer that originally transmitted it.

- When changing the ASA local IP address, you must ensure that all SXP peers have updated their peer list. In addition, if SXP peers changes its IP addresses, you must ensure those changes are reflected on the ASA.
- Automatic PAC file provisioning is not supported. The ASA administrator must request the PAC file from the ISE administrative interface and import it into the ASA.
- PAC files have expiration dates. You must import the updated PAC file before the current PAC file expires; otherwise, the ASA cannot retrieve environment data updates.
- When a security group changes on the ISE (for example, it is renamed or deleted), the ASA does not change the status of any ASA security policies that contain an SGT or security group name associated with the changed security group; however, the ASA generates a syslog message to indicate that those security policies changed.
- The multi-cast types are not supported in ISE 1.0.
- An SXP connection stays in the initializing state among two SXP peers interconnected by the ASA; as shown in the following example:

  (SXP peer A) - - - - (ASA) - - - (SXP peer B)

  Therefore, when configuring the ASA to integrate with Cisco TrustSec, you must enable the no-NAT, no-SEQ-RAND, and MD5-AUTHENTICATION TCP options on the ASA to configure SXP connections. Create a TCP state bypass policy for traffic destined to SXP port TCP 64999 among the SXP peers. Then apply the policy on the appropriate interfaces.

  For example, the following set of commands shows how to configure the ASA for a TCP state bypass policy:

```
access-list SXP-MD5-ACL extended permit tcp host peerA host peerB eq 64999
access-list SXP-MD5-ACL extended permit tcp host peerB host peerA eq 64999

tcp-map SXP-MD5-OPTION-ALLOW
 tcp-options range 19 19 allow

class-map SXP-MD5-CLASSMAP
 match access-list SXP-MD5-ACL

policy-map type inspect dns preset_dns_map
 parameters
  message-length maximum 512
policy-map global_policy
class SXP-MD5-CLASSMAP
  set connection random-sequence-number disable
  set connection advanced-options SXP-MD5-OPTION-ALLOW
  set connection advanced-options tcp-state-bypass
service-policy global_policy global
```

Configure the AAA Server for Cisco TrustSec Integration

This section describes how to integrate the AAA server for Cisco TrustSec.

Before You Begin

- The referenced server group must be configured to use the RADIUS protocol. If you add a non-RADIUS server group to the ASA, the configuration fails.
- If the ISE is also used for user authentication, obtain the shared secret that was entered on the ISE when you registered the ASA with the ISE. Contact your ISE administrator to obtain this information.

To configure the AAA server group to communicate with the ISE on the ASA, perform the following steps:

Procedure

Step 1 Create the AAA server group and configure the AAA server parameters for the ASA to communicate with the ISE server.

aaa-server *server-tag* **protocol radius**

Example:

```
hostname(config)# aaa-server ISEserver protocol radius
```

The *server-tag* argument specifies the server group name.

Step 2 Exit from the aaa server group configuration mode.

exit

Example:

```
hostname(config-aaa-server-group)# exit
```

Step 3 Configure a AAA server as part of a AAA server group and set host-specific connection data.

```
hostname(config)# aaa-server server-tag (interface-name) host server-ip
```

Example:

```
hostname(config)# aaa-server ISEserver (inside) host 192.0.2.1
```

The *interface-name* argument specifies the network interface where the ISE server resides. The parentheses are required in this parameter. The *server-tag* argument is the name of the AAA server group. The *server-ip* argument specifies the IP address of the ISE server.

Step 4 Specify the server secret value used to authenticate the ASA with the ISE server.

key *key*

Example:

```
hostname(config-aaa-server-host)# key myexclusivekey
```

The *key* argument is an alphanumeric keyword up to 127 characters long.

If the ISE is also used for user authentication, enter the shared secret that was entered on the ISE when you registered the ASA with the ISE.

Step 5 Exit from the aaa server host configuration mode.

exit

Example:

```
hostname(config-aaa-server-host)# exit
```

Step 6 Identify the AAA server group that is used by Cisco TrustSec for environment data retrieval.

cts server-group *AAA-server-group-name*

Example:

```
hostname(config)# cts server-group ISEserver
```

The *AAA-server-group-name* argument is the name of the AAA server group that you specified in Step 1 in the *server-tag* argument.

Note You may configure only one instance of the server group on the ASA for Cisco TrustSec.

Examples

The following example shows how to configure the ASA to communicate with the ISE server for Cisco TrustSec integration:

```
hostname(config)# aaa-server ISEserver protocol radius
hostname(config-aaa-server-group)# exit
hostname(config)# aaa-server ISEserver (inside) host 192.0.2.1
hostname(config-aaa-server-host)# key myexclusivemumblekey
hostname(config-aaa-server-host)# exit
hostname(config)# cts server-group ISEserver
```

Step 7

Import a PAC File

This section describes how to import a PAC file.

Before You Begin

- The ASA must be configured as a recognized Cisco TrustSec network device in the ISE before the ASA can generate a PAC file.
- Obtain the password used to encrypt the PAC file when generating it on the ISE. The ASA requires this password to import and decrypt the PAC file.
- The ASA requires access to the PAC file generated by the ISE. The ASA can import the PAC file from flash or from a remote server via TFTP, FTP, HTTP, HTTPS, or SMB. (The PAC file does not need to reside on the ASA flash before you can import it.)
- The server group has been configured for the ASA.

To import a PAC file, perform the following steps:

Procedure

Step 1 Import a Cisco TrustSec PAC file.

cts import-pac *filepath* **password** *value*

Example:

```
hostname(config)# cts import-pac disk0:/xyz.pac password IDFW-pac99
```

The *value* argument specifies the password used to encrypt the PAC file. The password is independent of the password that was configured on the ISE as part of the device credentials. The *filepath* argument is entered as one of the following options:

Single Mode

- **disk0**: Path and filename on disk0
- **disk1**: Path and filename on disk1
- **flash**: Path and filename on flash
- **ftp**: Path and filename on FTP
- **http**: Path and filename on HTTP
- **https**: Path and filename on HTTPS
- **smb**: Path and filename on SMB
- **tftp**: Path and filename on TFTP

Multi-mode

- **http**: Path and filename on HTTP
- **https**: Path and filename on HTTPS
- **smb**: Path and filename on SMB
- **tftp**: Path and filename on TFTP

Examples

The following example shows how to import a PAC file into the ASA:

```
hostname(config)# cts import pac disk0:/pac123.pac password hideme
PAC file successfully imported
```

The following example shows how to use the terminal to import a PAC file into the ASA:

```
hostname(config)# cts import-pac terminal password A9875Za551
Enter the PAC file data in ASCII hex format
End with the word "quit" on a line by itself.
hostname(exec_pac_hex)# 0100290405000001000000000000000
hostname(exec_pac_hex)# 00000000000000001111111111111111
hostname(exec_pac_hex)# 11111111111111112222222222222222
hostname(exec_pac_hex)# 222222222222222276d7d64b6be4804b
hostname(exec_pac_hex)# 0b4fdca3aeee11950ecd0e47c34157e5
hostname(exec_pac_hex)# 25f4964ed75835cde0adb7e198e0bcdb
hostname(exec_pac_hex)# 6aa8e363b0e4f9b4ac241be9ab576d0b
hostname(exec_pac_hex)# a1fcd34e5dd05dbe1312cbfea072fdb9
hostname(exec_pac_hex)# ee356fb61fe987d2d8f0ac3ef0467627
hostname(exec_pac_hex)# 7f8b137da2b840e16da520468b039bae
hostname(exec_pac_hex)# 36a4d844acc85cdefd7cb2cc58787590
hostname(exec_pac_hex)# ef123882a69b6c37bdbc9320e403024f
hostname(exec_pac_hex)# 354d42f404ec2d67ef3606575014584b
hostname(exec_pac_hex)# 2796e65ccd6e6c8d14d92448a8b24f6e
hostname(exec_pac_hex)# 47015a21f4f66cf6129d352bdfd4520f
hostname(exec_pac_hex)# 3f0c6f340a80715df4498956efe15dec
hostname(exec_pac_hex)# c08bb9a58cb6cb83ac91a3c40ce61de0
hostname(exec_pac_hex)# 284b743e52fd68e848685e2d78c33633
hostname(exec_pac_hex)# f2b4c5824138fc7bac9d9b83ac58ff9f
hostname(exec_pac_hex)# 1dbc84c416322f1f3c5951cf2132994a
hostname(exec_pac_hex)# a7cf20409df1d0d6621eba2b3af83252
```

```
hostname(exec_pac_hex)# 70d0130650122bdb13a83b2dae55533a
hostname(exec_pac_hex)# 4a394f21b441e164
hostname(exec_pac_hex)# quit
PAC Imported Successfully
hostname(config)#
```

-

Configure the Security Exchange Protocol

This section describes how to configure the Security Exchange Protocol.

Before You Begin

At least one interface must be in the UP/UP state.

Note When SXP is enabled with all interfaces down, the ASA does not display a message indicating that SXP is not working or it could not be enabled. If you check the configuration by entering the **show running-config** command, the command output displays the following message:

```
"WARNING: SXP configuration in process, please wait for a few moments and try again."
```

This message is generic and does not specify the reason why SXP is not working.

To configure SXP, perform the following steps:

Procedure

Step 1 Enable SXP on the ASA. By default, SXP is disabled.

cts sxp enable

Example:

```
hostname(config)# cts sxp enable
```

Step 2 Configure the default source IP address for SXP connections.

cts sxp default source-ip *ipaddress*

Example:

```
hostname(config)# cts sxp default source-ip 192.168.1.100
```

The *ipaddress* argument is an IPv4 or IPv6 address.

When you configure a default source IP address for SXP connections, you must specify the same address as the ASA outbound interface. If the source IP address does not match the address of the outbound interface, SXP connections fail.

When a source IP address for an SXP connection is not configured, the ASA performs a route/ARP lookup to determine the outbound interface for the SXP connection.

Step 3 Configure the default password for TCP MD5 authentication with SXP peers. By default, SXP connections do not have a password.

cts sxp default password [**0** | **8**] *password*

Example:

```
hostname(config)# cts sxp default password 8 IDFW-TrustSec-99
```

Configuring an encryption level for the password is optional. If you configure an encryption level, you can only set one level:

- Level 0—unencrypted cleartext
- Level 8—encrypted text

The *password* argument specifies an encrypted string of up to 162 characters or an ASCII key string up to 80 characters.

Step 4 Specify the default time interval between ASA attempts to set up new SXP connections between SXP peers.

cts sxp retry period *timervalue*

Example:

```
hostname(config)# cts sxp retry period 60
```

The ASA continues to make connection attempts until a successful connection is made. The retry timer is triggered as long as there is one SXP connection on the ASA that is not up.

The *timervalue* argument ranges from 0 to 64000 seconds. The default is 120 seconds. If you specify 0 seconds, the timer never expires and the ASA does not try to connect to SXP peers.

When the retry timer expires, the ASA goes through the connection database and if the database contains any connections that are off or in a "pending on" state, the ASA restarts the retry timer.

We recommend that you configure the retry timer to a different value from its SXP peer devices.

Step 5 Specify the value of the default reconcile timer.

cts sxp reconciliation period *timervalue*

Example:

```
hostname(config)# cts sxp reconciliation period 60
```

After an SXP peer terminates its SXP connection, the ASA starts a hold-down timer.

If an SXP peer connects while the hold-down timer is running, the ASA starts the reconcile timer; then the ASA updates the SXP mapping database to learn the latest mapping.

When the reconcile timer expires, the ASA scans the SXP mapping database to identify stale mapping entries (which were learned in a previous connection session). The ASA marks these connections as obsolete. When the reconcile timer expires, the ASA removes the obsolete entries from the SXP mapping database.

The *timervalue* argument ranges from 1 to 64000 seconds. The default is 120 seconds.

You cannot specify 0 seconds for the timer, because this value prevents the reconcile timer from starting. Not allowing the reconcile timer to run would keep stale entries for an undefined time and cause unexpected results from policy enforcement.

Examples

The following example shows how to set default values for SXP:

```
hostname(config)# cts sxp enable
hostname(config)# cts sxp default source-ip 192.168.1.100
hostname(config)# cts sxp default password 8 ********
```

```
hostname(config)# cts sxp retry period 60
hostname(config)# cts sxp reconcile period 60
```

Add an SXP Connection Peer

To add an SXP connection peer, perform the following steps:

Procedure

Step 1 Enable SXP on the ASA. By default, SXP is disabled.

cts sxp enable

Example:

```
hostname(config)# cts sxp enable
```

Step 2 Set up an SXP connection to an SXP peer.

cts sxp connection peer *peer_ip_address* [**source** *source_ip_address*] **password** {**default** | **none**} [**mode** {**local** | **peer**}] {**speaker** | **listener**}

Example:

```
hostname(config)# cts sxp connection peer 192.168.1.100 password default mode peer speaker
```

SXP connections are set per IP address; a single device pair can service multiple SXP connections.

The *peer_ip_address* argument is the IPv4 or IPv6 address of the SXP peer. The peer IP address must be reachable from the ASA outgoing interface.

The *source_ip_address* argument is the local IPv4 or IPv6 address of the SXP connection. The source IP address must be the same as the ASA outbound interface or the connection fails.

We recommend that you do not configure a source IP address for an SXP connection and allow the ASA to perform a route/ARP lookup to determine the source IP address for the SXP connection.

Indicate whether or not to use the authentication key for the SXP connection:

- **default**—Use the default password configured for SXP connections.
- **none**—Do not use a password for the SXP connection.

Indicate the mode of the SXP connection:

- **local**—Use the local SXP device.
- **peer**—Use the peer SXP device.

Indicate whether the ASA functions as a Speaker or Listener for the SXP connection.

- **speaker**— The ASA can forward IP-SGT mapping to upstream devices.
- **listener**—The ASA can receive IP-SGT mapping from downstream devices.

Examples

The following example shows how to configure SXP peers on the ASA:

```
hostname(config)# cts sxp enable
hostname(config)# cts sxp connection peer 192.168.1.100 password default mode peer speaker
hostname(config)# cts sxp connection peer 192.168.1.101 password default mode peer
```

```
hostname(config)# no cts sxp connection peer 192.168.1.100
hostname(config)# cts sxp connection peer 192.168.1.100 source 192.168.1.1 password default mode peer speaker
hostname(config)# no cts sxp connection peer 192.168.1.100 source 192.168.1.1 password default mode peer
speaker
```

Refresh Environment Data

The ASA downloads environment data from the ISE, which includes the Security Group Tag (SGT) name table. The ASA automatically refreshes its environment data that is obtained from the ISE when you complete the following tasks on the ASA:

- Configure a AAA server to communicate with the ISE.
- Import a PAC file from the ISE.
- Identify the AAA server group that the ASA will use to retrieve Cisco TrustSec environment data.

Normally, you do not need to manually refresh the environment data from the ISE; however, security groups can change on the ISE. These changes are not reflected on the ASA until you refresh the data in the ASA security group table, so refresh the data on the ASA to make sure that any security group changes made on the ISE are reflected on the ASA.

Note We recommend that you schedule policy configuration changes on the ISE and the manual data refresh on the ASA during a maintenance window. Handling policy configuration changes in this way maximizes the chances of security group names getting resolved and security policies becoming active immediately on the ASA.

To refresh the environment data, perform the following steps:

Procedure

Step 1 Refresh the environment data from the ISE and reset the reconcile timer to the configured default value.

```
cts refresh environment-data
```

Example:

```
hostname(config)# cts refresh environment-data
```

Configure the Security Policy

You can incorporate Cisco TrustSec policy in many ASA features. Any feature that uses extended ACLs (unless listed in this chapter as unsupported) can take advantage of Cisco TrustSec. You can add security group arguments to extended ACLs, as well as traditional network-based parameters.

- To configure an extended ACL, see the firewall configuration guide.
- To configure security group object groups that can be used in the ACL, see Configure Security Group Object Groups, page 2-8.

For example, an access rule permits or denies traffic on an interface using network information. With Cisco TrustSec, you can control access based on security group. For example, you could create an access rule for sample_securitygroup1 10.0.0.0 255.0.0.0, meaning the security group could have any IP address on subnet 10.0.0.0/8.

You can configure security policies based on combinations of security group names (servers, users, unmanaged devices, and so on), user-based attributes, and traditional IP-address-based objects (IP address, Active Directory object, and FQDN). Security group membership can extend beyond roles to include device and location attributes and is independent of user group membership.

Examples

The following example shows how to create an ACL that uses a locally defined security object group:

```
object-group security objgrp-it-admin
   security-group name it-admin-sg-name
   security-group tag 1
object-group security objgrp-hr-admin
   security-group name hr-admin-sg-name  // single sg_name
   group-object it-admin     // locally defined object-group as nested object
object-group security objgrp-hr-servers
    security-group name hr-servers-sg-name
object-group security objgrp-hr-network
    security-group tag 2
access-list hr-acl permit ip object-group-security objgrp-hr-admin any
object-group-security objgrp-hr-servers
```

The ACL configured in the previous example can be activated by configuring an access group or the Modular Policy Framework.

Additional examples:

```
!match src hr-admin-sg-name from any network to dst host 172.23.59.53
    access-list idw-acl permit ip security-group name hr-admin-sg-name any host 172.23.59.53
!match src hr-admin-sg-name from host 10.1.1.1 to dst any
    access-list idfw-acl permit ip security-group name hr-admin-sg-name host 10.1.1.1 any
!match src tag 22 from any network to dst hr-servers-sg-name any network
    access-list idfw-acl permit ip security-group tag 22 any security-group name hr-servers-sg-name any
!match src user mary from any host to dst hr-servers-sg-name any network
    access-list idfw-acl permit ip user CSCO\mary any security-group name hr-servers-sg-name any
!match src objgrp-hr-admin from any network to dst objgrp-hr-servers any network
    access-list idfw-acl permit ip object-group-security objgrp-hr-admin any object-group-security
    objgrp-hr-servers any
!match src user Jack from objgrp-hr-network and ip subnet 10.1.1.0/24 to dst objgrp-hr-servers any network
    access-list idfw-acl permit ip user CSCO\Jack object-group-security objgrp-hr-network 10.1.1.0
    255.255.255.0 object-group-security objgrp-hr-servers any
!match src user Tom from security-group mktg any google.com
object network net-google
    fqdn google.com
    access-list sgacl permit ip sec name mktg any object net-google
! If user Tom or object_group security objgrp-hr-admin needs to be matched, multiple ACEs can be defined as
follows:
    access-list idfw-acl2 permit ip user CSCO\Tom 10.1.1.0 255.255.255.0 object-group-security
    objgrp-hr-servers any
    access-list idfw-acl2 permit ip object-group-security objgrp-hr-admin 10.1.1.0 255.255.255.0
    object-group-security objgrp-hr-servers any
```

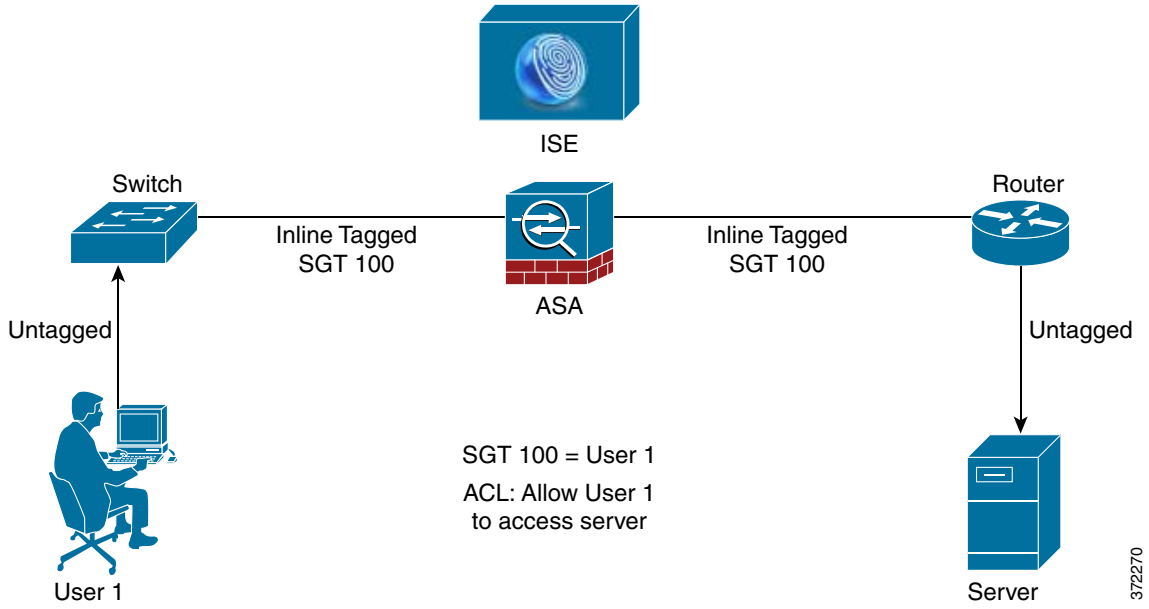
Layer 2 Security Group Tagging Imposition

Cisco TrustSec identifies and authenticates each network user and resource and assigns a 16-bit number called a Security Group Tag (SGT). This identifier is in turn propagated between network hops, which allows any intermediary devices such as ASAs, switches, and routers to enforce polices based on this identity tag.

SGT plus Ethernet Tagging, also called Layer 2 SGT Imposition, enables the ASA to send and receive security group tags on Ethernet interfaces using Cisco proprietary Ethernet framing (EtherType 0x8909), which allows the insertion of source security group tags into plain-text Ethernet frames. The ASA inserts security group tags on the outgoing packet and processes security group tags on the incoming packet, based on a manual per-interface configuration. This feature allows inline hop-by-hop propagation of endpoint identity across network devices and provides seamless Layer 2 SGT Imposition between each hop.

The following figure shows a typical example of Layer 2 SGT Imposition.

Figure 6-3 Layer 2 SGT Imposition



Usage Scenarios

The following table describes the expected behavior for ingress traffic when configuring this feature.

Table 6-3 Ingress Traffic

Interface Configuration	Tagged Packet Received	Untagged Packet Received
No command is issued.	Packet is dropped.	SGT value is from the IP-SGT Manager.
The **cts manual** command is issued.	SGT value is from the IP-SGT Manager.	SGT value is from the IP-SGT Manager.

Table 6-3 Ingress Traffic

Interface Configuration	Tagged Packet Received	Untagged Packet Received
The **cts manual** command and the **policy static sgt** *sgt_number* command are both issued.	SGT value is from the **policy static sgt** *sgt_number* command.	SGT value is from the **policy static sgt** *sgt_number* command.
The **cts manual** command and the **policy static sgt** *sgt_number* **trusted** command are both issued.	SGT value is from the inline SGT in the packet.	SGT value is from the **policy static sgt** *sgt_number* command.

Note If there is no matched IP-SGT mapping from the IP-SGT Manager, then a reserved SGT value of "0x0" for "Unknown" is used.

The following table describes the expected behavior for egress traffic when configuring this feature.

Table 6-4 Egress Traffic

Interface Configuration	Tagged or Untagged Packet Sent
No command is issued.	Untagged
The **cts manual** command is issued.	Tagged
The **cts manual** command and the **propagate sgt** command are both issued.	Tagged
The **cts manual** command and the **no propagate sgt** command are both issued.	Untagged

The following table describes the expected behavior for to-the-box and from-the-box traffic when configuring this feature.

Table 6-5 To-the-box and From-the-box Traffic

Interface Configuration	Tagged or Untagged Packet Received
No command is issued on the ingress interface for to-the-box traffic.	Packet is dropped.
The **cts manual** command is issued on the ingress interface for to-the-box traffic.	Packet is accepted, but there is no policy enforcement or SGT propagation.
The **cts manual** command is not issued or the **cts manual** command and **no propagate sgt** command are both issued on the egress interface for from-the-box traffic.	Untagged packet is sent, but there is no policy enforcement. The SGT number is from the IP-SGT Manager.
The **cts manual** command is issued or the **cts manual** command and the **propagate sgt** command are both issued on the egress interface for from-the-box traffic.	Tagged packet is sent. The SGT number is from the IP-SGT Manager.

Note If there is no matched IP-SGT mapping from the IP-SGT Manager, then a reserved SGT value of "0x0" for "Unknown" is used.

Configure a Security Group Tag on an Interface

To configure a security group tag on an interface, perform the following steps:

Procedure

Step 1 Specify an interface and enter interface configuration mode.

interface *id*

Example:

```
ciscoasa(config)# interface gi0/0
```

Step 2 Enable Layer 2 SGT Imposition and enter cts manual interface configuration mode.

cts manual

Example:

```
hostname(config-if)# cts manual
```

Step 3 Enable propagation of a security group tag on an interface. Propagation is enabled by default.

propagate sgt

Example:

```
hostname(config-if-cts-manual)# propagate sgt
```

Step 4 Apply a policy to a manually configured CTS link.

policy static sgt *sgt_number* [**trusted**]

Example:

```
hostname(config-if-cts-manual)# policy static sgt 50 trusted
```

The **static** keyword specifies an SGT policy to incoming traffic on the link.

The **sgt** *sgt_number* keyword-argument pair specifies the SGT number to apply to incoming traffic from the peer. Valid values are from 2-65519.

The **trusted** keyword indicates that ingress traffic on the interface with the SGT specified in the command should not have its SGT overwritten. Untrusted is the default.

Examples

The following example enables an interface for Layer 2 SGT imposition and defines whether or not the interface is trusted:

```
ciscoasa(config)# interface gi0/0
ciscoasa(config-if)# cts manual
ciscoasa(config-if-cts-manual)# propagate sgt
ciscoasa(config-if-cts-manual)# policy static sgt 50 trusted
```

Configure IP-SGT Bindings Manually

To configure IP-SGT bindings manually, perform the following steps:

Procedure

Step 1 Configure IP-SGT bindings manually.

cts role-based sgt-map [*IPv4_addr* | *IPv6_addr*] **sgt** *sgt_value*

Example:

```
hostname(config)# cts role-based sgt-map 10.2.1.2 sgt 50
```

The **sgt** *sgt_value* keyword-argument pair specifies the SGT number. Valid values are from 2-65519.

Troubleshooting Tips

Use the **packet-tracer** command to determine why a particular session was allowed or denied, which SGT value is being used (from the SGT in the packet, from the IP-SGT manager, or from the **policy static sgt** command configured on the interface), and which security group-based security policies were applied.

The following example displays output from the **packet-tracer** command to show security group tag mapping to an IP address:

```
hostname# packet-tracer input inside tcp inline-tag 100 security-group name alpha 30
security-group tag 31 300
Mapping security-group 30:alpha to IP address 10.1.1.2.
Mapping security-group 31:bravo to IP address 192.168.1.2.

Phase: 1
Type: ROUTE-LOOKUP
Subtype: input
Result: ALLOW
Config:
Additional Information:
in 192.168.1.0 255.255.255.0 outside....
----------------More---------------------
```

Use the **capture** *capture-name* **type inline-tag** *tag* command to capture only the Cisco CMD packets (EtherType 0x8909) with or without a specific SGT value.

The following example displays output from the **show capture** command for a specified SGT value:

```
hostname# show capture my-inside-capture
1: 11:34:42.931012 INLINE-TAG 36 10.0.101.22 > 10.0.101.100: icmp: echo request
2: 11:34:42.931470 INLINE-TAG 48 10.0.101.100 > 10.0.101.22: icmp: echo reply
3: 11:34:43.932553 INLINE-TAG 36 10.0.101.22 > 10.0.101.100: icmp: echo request
4: 11.34.43.933164 INLINE-TAG 48 10.0.101.100 > 10.0.101.22: icmp: echo reply
```

Example for Cisco TrustSec

The following example shows how to configure the ASA to use Cisco TrustSec:

```
// Import an encrypted CTS PAC file
    cts import-pac asa.pac password Cisco
// Configure ISE for environment data download
    aaa-server cts-server-list protocol radius
    aaa-server cts-server-list host 10.1.1.100 cisco123
    cts server-group cts-server-list
// Configure SXP peers
    cts sxp enable
    cts sxp connection peer 192.168.1.100 password default mode peer speaker
//Configure security-group based policies
    object-group security objgrp-it-admin
          security-group name it-admin-sg-name
          security-group tag 1
    object-group security objgrp-hr-admin
        security-group name hr-admin-sg-name
        group-object it-admin
    object-group security objgrp-hr-servers
        security-group name hr-servers-sg-name
    access-list hr-acl permit ip object-group-security objgrp-hr-admin any
    object-group-security objgrp-hr-servers
//Configure security group tagging plus Ethernet tagging
   interface gi0/1
   cts manual
   propagate sgt
   policy static sgt 100 trusted
   cts role-based sgt-map 10.1.1.100 sgt 50
```

AnyConnect VPN Support for Cisco TrustSec

ASA Version 9.3(1) fully supports security group tagging of VPN sessions. A Security Group Tag (SGT) can be assigned to a VPN session using an external AAA server, or by configuration of the local user database. This tag can then be propagated through the Cisco TrustSec system over Layer 2 Ethernet. Security group tags are useful on group policies and for local users when the AAA server cannot provide an SGT.

If there is no SGT in the attributes from the AAA server to assign to a VPN user, then the ASA uses the SGT in the default group policy. If there is no SGT in the group policy, then tag 0x0 is assigned.

Typical Steps for a Remote User Connecting to a Server

1. A user connects to the ASA.
2. The ASA requests AAA information from the ISE, which may include an SGT. The ASA also assigns an IP address for the user's tunneled traffic.
3. The ASA uses AAA information to authenticate and creates a tunnel.
4. The ASA uses the SGT from AAA information and the assigned IP address to add an SGT in the Layer 2 header.
5. Packets that include the SGT are passed to the next peer device in the Cisco TrustSec network.

Add an SGT to Local Users and Groups

To configure an SGT attribute on the LOCAL user database and in a group policy, perform the following steps:

Procedure

Step 1 Enter group-policy configuration mode.

group-policy *name*

Example:

```
hostname(config)# group policy Grpolicy1
```

Step 2 Configure SGT attributes on the named group policy's or LOCAL username's attribute set.

security-group-tag value *sgt*

Example:

```
hostname(config-group-policy# security-group-tag value 101
```

The default form of this command is **security-group-tag none**, which means that there is no security group tag in this attribute set. Use the **no security-group-tag value** *sgt* command to return the configuration to the default.

Monitoring Cisco TrustSec

See the following commands for monitoring Cisco TrustSec:

- **show running-config cts**
- **show running-config** [**all**] **cts role-based** [**sgt-map**]

  This command shows the user-defined IP-SGT binding table entries.
- **show cts sxp connections**

  This command shows the SXP connections on the ASA for a particular user context when multiple context mode is used.
- **show conn security-group**

  Shows data for all SXP connections.
- **show cts environment-data**

  Shows the Cisco TrustSec environment information contained in the security group table on the ASA.
- **show cts sgt-map**

  Shows the IP address-security group table manager entries in the control path.
- **show asp table cts sgt-map**

  This command shows the IP address-security group table mapping entries from the IP address-security group table mapping database maintained in the datapath.

- **show cts pac**

History for Cisco TrustSec

Table 6-6 History for Cisco TrustSec

Feature Name	Platform Releases	Description
Cisco TrustSec	9.0(1)	Cisco TrustSec provides access control that builds on an existing identity-aware infrastructure to ensure data confidentiality between network devices and integrate security access services on one platform. In the Cisco TrustSec feature, enforcement devices use a combination of user attributes and endpoint attributes to make role-based and identity-based access control decisions. In this release, the ASA integrates with Cisco TrustSec to provide security group-based policy enforcement. Access policies within the Cisco TrustSec domain are topology-independent, based on the roles of source and destination devices rather than on network IP addresses. The ASA can use Cisco TrustSec for other types of security group-based policies, such as application inspection; for example, you can configure a class map that includes an access policy based on a security group. We introduced or modified the following commands: **access-list extended**, **cts sxp enable**, **cts server-group**, **cts sxp default**, **cts sxp retry period**, **cts sxp reconciliation period**, **cts sxp connection peer**, **cts import-pac**, **cts refresh environment-data**, **object-group security**, **security-group**, **show running-config cts**, **show running-config object-group**, **clear configure cts**, **clear configure object-group**, **show cts pac**, **show cts environment-data**, **show cts environment-data sg-table**, **show cts sxp connections**, **show object-group**, **show configure security-group**, **clear cts environment-data**, **debug cts,** and **packet-tracer**.
Layer 2 Security Group Tag Imposition	9.3(1)	You can now use security group tagging combined with Ethernet tagging to enforce policies. SGT plus Ethernet Tagging, also called Layer 2 SGT Imposition, enables the ASA to send and receive security group tags on Ethernet interfaces using Cisco proprietary Ethernet framing (EtherType 0x8909), which allows the insertion of source security group tags into plain-text Ethernet frames. We introduced or modified the following commands: **cts manual**, **policy static sgt**, **propagate sgt**, **cts role-based sgt-map**, **show cts sgt-map**, **packet-tracer**, **capture**, **show capture**, **show asp drop**, **show asp table classify**, **show running-config all**, **clear configure all**, and **write memory**.

ASA FirePOWER Module

This chapter describes how to configure the ASA FirePOWER module that runs on the ASA.

About the ASA FirePOWER Module

The ASA FirePOWER module supplies next-generation firewall services, including Next-Generation Intrusion Prevention System (NGIPS), Application Visibility and Control (AVC), URL filtering, and Advanced Malware Protection (AMP).

The ASA FirePOWER module runs a separate application from the ASA. The module can be a hardware module (on the ASA 5585-X only) or a software module (all other models).

How the ASA FirePOWER Module Works with the ASA

You can configure your ASA FirePOWER module using one of the following deployment models:

- Inline mode—In an inline deployment, the actual traffic is sent to the ASA FirePOWER module, and the module's policy affects what happens to the traffic. After dropping undesired traffic and taking any other actions applied by policy, the traffic is returned to the ASA for further processing and ultimate transmission.

- Inline tap monitor-only mode (ASA inline)—In an inline tap monitor-only deployment, a copy of the traffic is sent to the ASA FirePOWER module, but it is not returned to the ASA. Inline tap mode lets you see what the ASA FirePOWER module would have done to traffic, and lets you evaluate the content of the traffic, without impacting the network. However, in this mode, the ASA does apply its policies to the traffic, so traffic can be dropped due to access rules, TCP normalization, and so forth.
- Passive monitor-only (traffic forwarding) mode—If you want to prevent any possibility of the ASA with FirePOWER Services device impacting traffic, you can configure a traffic-forwarding interface and connect it to a SPAN port on a switch. In this mode, traffic is sent directly to the ASA FirePOWER module without ASA processing. The traffic is "black holed," in that nothing is returned from the module, nor does the ASA send the traffic out any interface. You must operate the ASA in single context transparent mode to configure traffic forwarding.

Be sure to configure consistent policies on the ASA and the ASA FirePOWER. Both policies should reflect the inline or monitor-only mode of the traffic.

The following sections explain these modes in more detail.

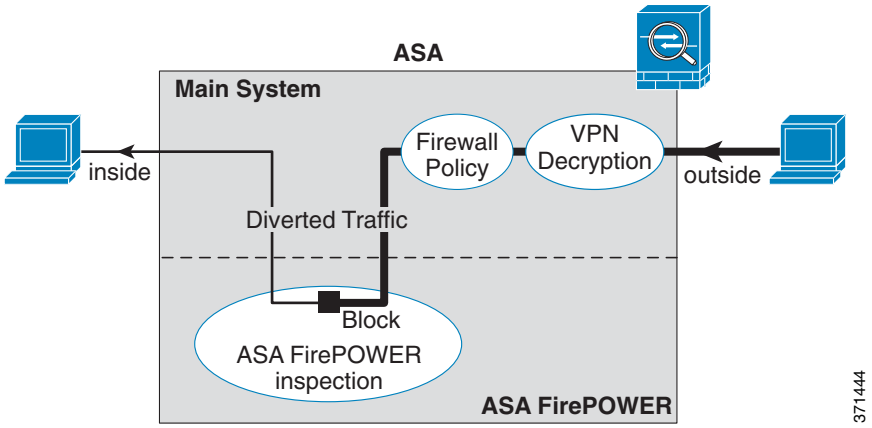
ASA FirePOWER Inline Mode

In inline mode, traffic goes through the firewall checks before being forwarded to the ASA FirePOWER module. When you identify traffic for ASA FirePOWER inspection on the ASA, traffic flows through the ASA and the module as follows:

1. Traffic enters the ASA.
2. Incoming VPN traffic is decrypted.
3. Firewall policies are applied.
4. Traffic is sent to the ASA FirePOWER module.
5. The ASA FirePOWER module applies its security policy to the traffic, and takes appropriate actions.
6. Valid traffic is sent back to the ASA; the ASA FirePOWER module might block some traffic according to its security policy, and that traffic is not passed on.
7. Outgoing VPN traffic is encrypted.
8. Traffic exits the ASA.

The following figure shows the traffic flow when using the ASA FirePOWER module in inline mode. In this example, the module blocks traffic that is not allowed for a certain application. All other traffic is forwarded through the ASA.

Figure 7-1 ASA FirePOWER Module Traffic Flow in the ASA



Note If you have a connection between hosts on two ASA interfaces, and the ASA FirePOWER service policy is only configured for one of the interfaces, then all traffic between these hosts is sent to the ASA FirePOWER module, including traffic originating on the non-ASA FirePOWER interface (because the feature is bidirectional).

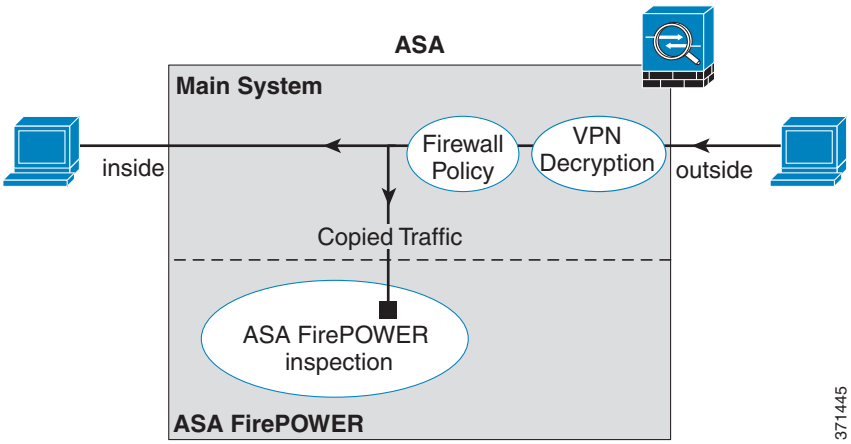
ASA FirePOWER Inline Tap Monitor-Only Mode

This mode sends a duplicate stream of traffic to the ASA FirePOWER module for monitoring purposes only. The module applies the security policy to the traffic and lets you know what it would have done if it were operating in inline mode; for example, traffic might be marked “would have dropped” in events. You can use this information for traffic analysis and to help you decide if inline mode is desirable.

Note You cannot configure both inline tap monitor-only mode and normal inline mode at the same time on the ASA. Only one type of security policy is allowed. In multiple context mode, you cannot configure inline tap monitor-only mode for some contexts, and regular inline mode for others.

The following figure shows the traffic flow when operating in inline tap mode.

Figure 7-2 ASA FirePOWER Inline Tap Monitor-Only Mode



ASA FirePOWER Passive Monitor-Only Traffic Forwarding Mode

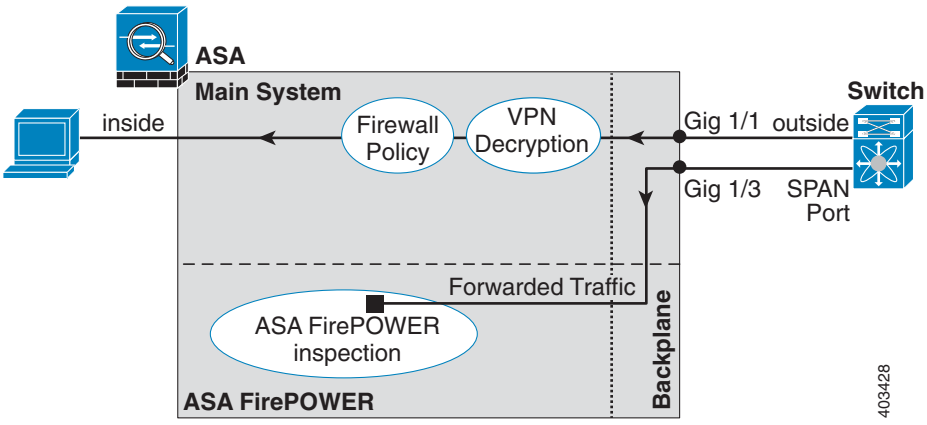
If you want to operate the ASA FirePOWER module as a pure Intrusion Detection System (IDS), where there is no impact on the traffic at all, you can configure a traffic forwarding interface. A traffic forwarding interface sends all received traffic directly to the ASA FirePOWER module without any ASA processing.

The module applies the security policy to the traffic and lets you know what it would have done if it were operating in inline mode; for example, traffic might be marked “would have dropped” in events. You can use this information for traffic analysis and to help you decide if inline mode is desirable.

Traffic in this setup is never forwarded: neither the module nor the ASA sends the traffic on to its ultimate destination. You must operate the ASA in single context and transparent modes to use this configuration.

The following figure shows an interface configured for traffic-forwarding. That interface is connected to a switch SPAN port so the ASA FirePOWER module can inspect all of the network traffic. Another interface sends traffic normally through the firewall.

Figure 7-3 ASA FirePOWER Passive Monitor-Only, Traffic-Forwarding Mode



ASA FirePOWER Management

The module has a basic command line interface (CLI) for initial configuration and troubleshooting only. You configure the security policy on the ASA FirePOWER module using one of the following methods:

- FireSIGHT Management Center (all models)—Can be hosted on a separate FireSIGHT Management Center appliance or as a virtual appliance.
- ASDM (ASA 5506-X, 5508-X, and 5516-X)—You can manage both the ASA and the module using the on-box ASDM.

Compatibility with ASA Features

The ASA includes many advanced application inspection features, including HTTP inspection. However, the ASA FirePOWER module provides more advanced HTTP inspection than the ASA provides, as well as additional features for other applications, including monitoring and controlling application usage.

To take full advantage of the ASA FirePOWER module features, use the following guidelines for traffic that you send to the ASA FirePOWER module:

- Do not configure ASA inspection on HTTP traffic.
- Do not configure Cloud Web Security (ScanSafe) inspection. If you configure both ASA FirePOWER inspection and Cloud Web Security inspection for the same traffic, the ASA only performs ASA FirePOWER inspection.
- Do not enable the Mobile User Security (MUS) server; it is not compatible with the ASA FirePOWER module.

Other application inspections on the ASA are compatible with the ASA FirePOWER module, including the default inspections.

Licensing Requirements for the ASA FirePOWER Module

The ASA FirePOWER module and FireSIGHT Management Center require additional licenses, which need to be installed in the module itself rather than the ASA. The ASA itself requires no additional licenses.

See the Licensing chapter of the *FireSIGHT System User Guide* or the online help in FireSIGHT Management Center for more information.

Guidelines for ASA FirePOWER

Failover Guidelines

- Does not support failover directly; when the ASA fails over, any existing ASA FirePOWER flows are transferred to the new ASA. The ASA FirePOWER module in the new ASA begins inspecting the traffic from that point forward; old inspection states are not transferred.
- You are responsible for maintaining consistent policies on the ASA FirePOWER modules in the high-availability ASA pair to ensure consistent failover behavior.

ASA Clustering Guidelines

Does not support clustering directly, but you can use these modules in a cluster. You are responsible for maintaining consistent policies on the ASA FirePOWER modules in the cluster using FireSIGHT Management Center.

Model Guidelines

- For ASA model software and hardware compatibility with the ASA FirePOWER module, see Cisco ASA Compatibility.
- For the 5512-X through ASA 5555-X, you must install a Cisco solid state drive (SSD). For more information, see the ASA 5500-X hardware guide. (The SSD is standard on the 5506-X, 5508-X, and 5516-X.)

ASDM Guidelines for Managing ASA FirePOWER (Supported Models)

- If you enable command authorization on the ASA that hosts the module, you must log in with a user name that has privilege level 15 to see the **ASA FirePOWER** home, configuration, and monitoring pages. Read-only or monitor-only access to **ASA FirePOWER** pages other than the status page is not supported.
- If you are using Java 7 update 51 up to Java 8, you need to configure identity certificates for both the ASA and the ASA FirePOWER module. See Install an Identity Certificate for ASDM.
- You can never use both ASDM and FireSIGHT Management Center, you must choose one or the other.

Additional Guidelines and Limitations

- See Compatibility with ASA Features, page 7-5.
- You cannot change the software type installed on the hardware module; if you purchase an ASA FirePOWER module, you cannot later install other software on it.
- You cannot configure both normal inline mode and inline tap monitor-only mode at the same time on the ASA. Only one type of security policy is allowed. In multiple context mode, you cannot configure inline tap monitor-only mode for some contexts, and regular inline mode for others.

Defaults for ASA FirePOWER

The following table lists the default settings for the ASA FirePOWER module.

Table 7-1 ASA FirePOWER Default Network Parameters

Parameters	Default
Management IP address	• System software image: 192.168.45.45/24 • Boot image: 192.168.8.8/24
Gateway	• System software image: none • Boot image: 192.168.8.1/24
SSH or session Username	**admin**
Password	• System software image: **Sourcefire** • Boot image: **Admin123**

Perform Initial ASA FirePOWER Setup

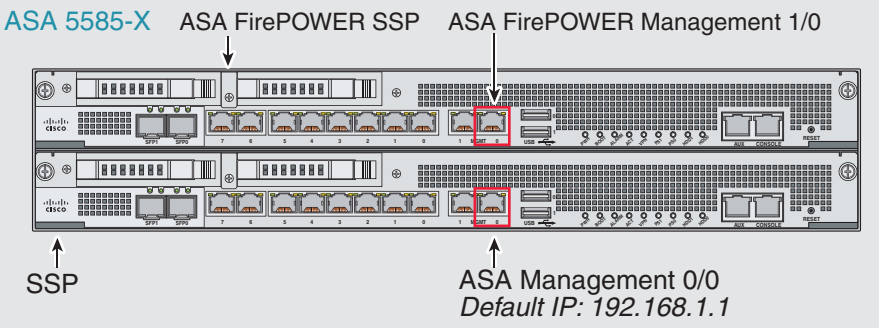
Deploy the ASA FirePOWER module in your network, and then access the ASA FirePOWER CLI to perform initial setup.

Deploy the ASA FirePOWER Module in Your Network

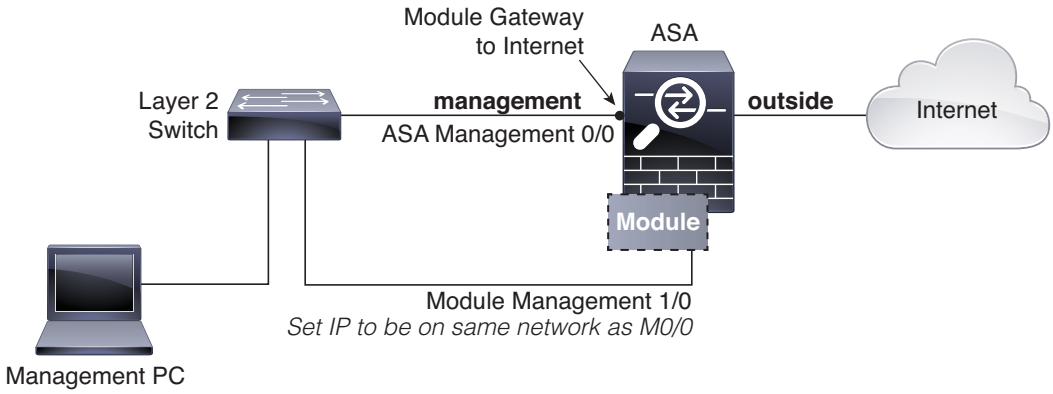
See the section for your ASA model to determine how to connect the ASA FirePOWER module management interface to your network.

ASA 5585-X (Hardware Module)

The ASA FirePOWER module includes separate management interfaces from the ASA.



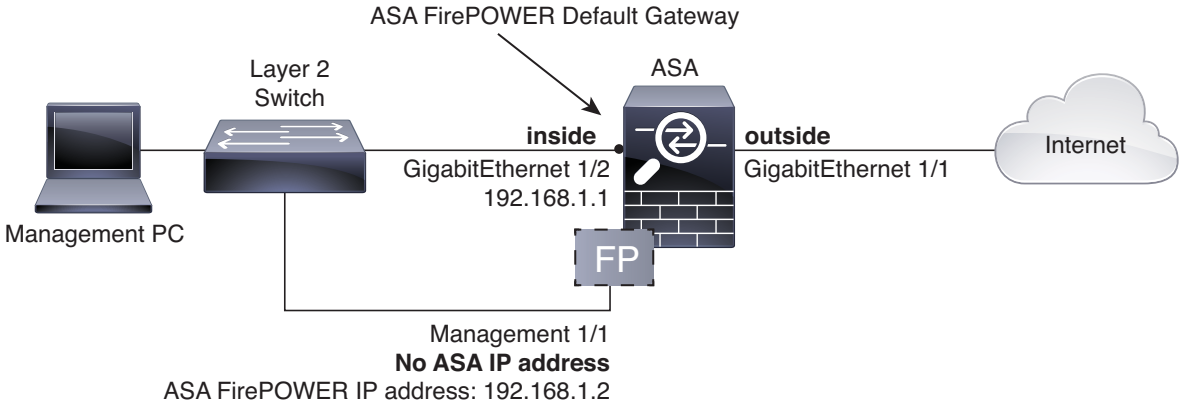
All management traffic to and from the ASA FirePOWER module must enter and exit the Management 1/0 or 1/1 interface. Because this interface is not an ASA data interface, traffic cannot pass through the ASA over the backplane; you need to physically cable the management interface to an ASA interface. The ASA FirePOWER module needs Internet access. See the following typical cabling setup to allow ASA FirePOWER access to the Internet through the ASA management interface. Other options are possible, depending on how you want to connect your network; for example, you can make the Management 1/0 interface outside facing; or you can route between it and a different ASA interface if you have an inside router.



ASA 5506-X through ASA 5555-X (Software Module)

These models run the ASA FirePOWER module as a software module, and the ASA FirePOWER management interface shares the Management 0/0 or Management 1/1 interface (depending on your model) with the ASA.

The following figure shows the recommended network deployment for the ASA 5500-X with the ASA FirePOWER module:



For the ASA 5506-X, 5508-X, and 5516-X, the default configuration enables the above network deployment; the only change you need to make is to set the module IP address to be on the same network as the ASA inside interface and to configure the module gateway IP address.

For other models, you must remove the ASA-configured name and IP address for Management 0/0 or 1/1, and then configure the other interfaces as indicated above.

Note If you want to deploy a separate router on the inside network, then you can route between management and inside. In this case, you can manage both the ASA and ASA FirePOWER module on the Management interface with the appropriate configuration changes.

Access the ASA FirePOWER CLI

To access the ASA FirePOWER CLI, you can use one of the following methods.

Console Port

- ASA 5585-X—This model includes a dedicated console port for the ASA FirePOWER module. Use the supplied DB-9 to RJ-45 serial cable and/or your own USB serial adapter.
- All other models—Connect to the ASA console port using the supplied DB-9 to RJ-45 serial cable and/or your own USB serial adapter. The ASA 5506-X/5508-X/5516-X also has a mini-USB console port. See the hardware guide for instructions on using the USB console port.

  At the ASA CLI, session to the ASA FirePOWER module:

  ```
  session sfr
  ```

  See also Session to the Software Module From the ASA, page 7-20.

SSH

You can connect to the module default IP address (see Defaults for ASA FirePOWER, page 7-6) or you can use the following ASA command to change the management IP address, and then connect using SSH:

```
session {1 | sfr} do setup host ip ip_address/mask,gateway_ip
```

Use **1** for a hardware module, **sfr** for a software module.

Configure ASA FirePOWER Basic Settings

The first time you access the ASA FirePOWER module CLI, you are prompted for basic configuration parameters. You must also add the module to the FireSIGHT Management Center (optional for the ASA 5506-X/5508-X/5516-X).

Procedure

Step 1 At the ASA FirePOWER CLI, log in with the username **admin** and the password **Sourcefire**.

Step 2 Complete the system configuration as prompted.

Use the following network settings for the ASA FirePOWER module for the recommended network deployment (Deploy the ASA FirePOWER Module in Your Network, page 7-7):

- Management interface: 192.168.1.2
- Management subnet mask: 255.255.255.0
- Gateway IP: 192.168.1.1

Step 3 (Optional for 5506-X/5508-X/5516-X) Register the ASA FirePOWER module to a FireSIGHT Management Center:

```
> configure manager add {hostname | IPv4_address | IPv6_address | DONTRESOLVE} reg_key
[nat_id]
```

where:

- {*hostname* | *IPv4_address* | *IPv6_address* | **DONTRESOLVE**} specifies either the fully qualified host name or IP address of the FireSIGHT Management Center. If the FireSIGHT Management Center is not directly addressable, use DONTRESOLVE.
- *reg_key* is the unique alphanumeric registration key required to register a ASA FirePOWER module to the FireSIGHT Management Center.
- *nat_id* is an optional alphanumeric string used during the registration process between the FireSIGHT Management Center and the ASA FirePOWER module. It is required if the hostname is set to DONTRESOLVE.

Step 4 Close the console connection. For the software module, enter:

```
> exit
```

Configure the ASA FirePOWER Module

Configure the security policy in the ASA FirePOWER OS, and then configure the ASA to send traffic to the module.

- Configure the Security Policy on the ASA FirePOWER Module, page 7-10
- Redirect Traffic to the ASA FirePOWER Module, page 7-10

Configure the Security Policy on the ASA FirePOWER Module

The security policy controls the services provided by the module, such as Next Generation IPS filtering and application filtering. You configure the security policy on the ASA FirePOWER module using one of the following methods.

For more information about ASA FirePOWER configuration, see the online help or the ASA FirePOWER Module User Guide or FireSIGHT System User Guide.

FireSIGHT Management Center (All Models)

Use a web browser to open **https://***DC_address*, where *DC_address* is the DNS name or IP address of the manager you defined in Configure ASA FirePOWER Basic Settings, page 7-9. For example, https://dc.example.com.

Alternatively, in ASDM, choose **Home > ASA FirePOWER Status** and click the link at the bottom of the dashboard.

ASDM (ASA 5506-X, 5508-X, and 5516-X)

In ASDM, choose **Configuration > ASA FirePOWER Configuration**.

Redirect Traffic to the ASA FirePOWER Module

For inline and inline tap (monitor-only) modes, you configure a service policy to redirect traffic to the module. If you want passive monitor-only mode, you configure a traffic redirection interface, which bypasses ASA policies.

The following topics explain how to configure these modes.

Configure Inline or Inline Tap Monitor-Only Modes

Redirect traffic to the ASA FirePOWER module by creating a service policy that identifies specific traffic that you want to send. In this mode, ASA policies, such as access rules, are applied to the traffic before it is redirected to the module.

Before You Begin

- If you have an active service policy redirecting traffic to an IPS or CX module (that you replaced with ASA FirePOWER), you must remove that policy before you configure the ASA FirePOWER service policy.
- Be sure to configure consistent policies on the ASA and the ASA FirePOWER. Both policies should reflect the inline or inline tap mode of the traffic.
- In multiple context mode, perform this procedure within each security context.

Procedure

Step 1 Create an L3/L4 class map to identify the traffic that you want to send to the module.

```
class-map name
   match parameter
```

Example:

```
hostname(config)# access-list my-sfr-acl permit ip any 10.1.1.0 255.255.255.0
hostname(config)# access-list my-sfr-acl2 permit ip any 10.2.1.0 255.255.255.0
hostname(config)# class-map my-sfr-class
hostname(config-cmap)# match access-list my-sfr-acl
```

If you want to send multiple traffic classes to the module, you can create multiple class maps for use in the security policy. For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

```
policy-map {global_policy | name}
```

Example:

```
hostname(config)# policy-map inside_policy
```

In the default configuration, the **global_policy** policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter **global_policy** as the policy name. To create a new interface-based policy, specify a new name.

Step 3 Identify the class map you created at the start of this procedure.

```
class name
```

Example:

```
hostname(config-pmap)# class my-sfr-class
```

Step 4 Send the traffic to the ASA FirePOWER module.

```
sfr {fail-close | fail-open} [monitor-only]
```

Where:

- The **fail-close** keyword sets the ASA to block all traffic if the ASA FirePOWER module is unavailable.

- The **fail-open** keyword sets the ASA to allow all traffic through, uninspected, if the module is unavailable.
- Specify **monitor-only** to send a read-only copy of traffic to the module, i.e. inline tap mode. If you do not include the keyword, the traffic is sent in inline mode. Be sure to configure consistent policies on the ASA and the ASA FirePOWER. See ASA FirePOWER Inline Tap Monitor-Only Mode, page 7-3 for more information.

Example:

```
hostname(config-pmap-c)# sfr fail-close
```

Step 5 If you created multiple class maps for ASA FirePOWER traffic, you can specify another class for the policy and apply the **sfr** redirect action.

See Feature Matching Within a Service Policy, page 11-5 for detailed information about how the order of classes matters within a policy map. Traffic cannot match more than one class map for the same action type.

Step 6 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy inside_policy interface inside
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Configure Passive Traffic Forwarding

If you want to operate the module in passive monitor-only mode, where the module gets a copy of the traffic and neither it nor the ASA can affect the network, configure a traffic forwarding interface and connect the interface to a SPAN port on a switch. For more details, see ASA FirePOWER Passive Monitor-Only Traffic Forwarding Mode, page 7-4.

The following guidelines explain the requirements for this deployment mode:

- The ASA must be in single-context and transparent mode.
- You can configure up to 4 interfaces as traffic-forwarding interfaces. Other ASA interfaces can be used as normal.
- Traffic-forwarding interfaces must be physical interfaces, not VLANs or BVIs. The physical interface also cannot have any VLANs associated with it.
- Traffic-forwarding interfaces cannot be used for ASA traffic; you cannot name them or configure them for ASA features, including failover or management-only.
- You cannot configure both a traffic-forwarding interface and a service policy for ASA FirePOWER traffic.

Procedure

Step 1 Enter interface configuration mode for the physical interface you want to use for traffic-forwarding.

interface *physical_interface*

Example:

```
hostname(config)# interface gigabitethernet 0/5
```

Step 2 Remove any name configured for the interface. If this interface was used in any ASA configuration, that configuration is removed. You cannot configure traffic-forwarding on a named interface.

```
no nameif
```

Step 3 Enable traffic-forwarding.

```
traffic-forward sfr monitor-only
```

Note You can ignore any warnings about traffic forwarding being for demonstration purposes only. This is a supported production mode.

Step 4 Enable the interface.

```
no shutdown
```

Repeat for any additional interfaces.

Examples

The following example makes GigabitEthernet 0/5 a traffic-forwarding interface:

```
interface gigabitethernet 0/5
  no nameif
  traffic-forward sfr monitor-only
  no shutdown
```

Managing the ASA FirePOWER Module

This section includes procedures that help you manage the module.

Install or Reimage the Module

This section describes how to install or reimage a software or hardware module.

- Reimage the ASA 5585-X ASA FirePOWER Hardware Module, page 7-16

Install or Reimage the Software Module

If you purchase the ASA with the ASA FirePOWER module, the module software and required solid state drives (SSDs) come pre-installed and ready to configure. If you want to add the ASA FirePOWER software module to an existing ASA, or need to replace the SSD, you need to install the ASA FirePOWER boot software, partition the SSD, and install the system software according to this procedure.

Reimaging the module is the same procedure, except you should first uninstall the ASA FirePOWER module. You would reimage a system if you replace an SSD.

For information on how to physically install the SSD, see the ASA hardware guide.

Before You Begin

- The free space on flash (disk0) should be at least 3GB plus the size of the boot software.
- In multiple context mode, perform this procedure in the system execution space.
- You must shut down any other software module that you might be running; the ASA can run a single software module at a time. You must do this from the ASA CLI. For example, the following commands shut down and uninstall the IPS software module, and then reload the ASA; the commands to remove the CX module are the same, except use the **cxsc** keyword instead of **ips**.

  ```
  sw-module module ips shutdown
  sw-module module ips uninstall
  reload
  ```

  When reimaging the ASA FirePOWER module, use the same shutdown and uninstall commands to remove the old image. For example, **sw-module module sfr uninstall**.
- If you have an active service policy redirecting traffic to an IPS or CX module, you must remove that policy. For example, if the policy is a global one, you could use **no service-policy** *ips_policy* **global**. If the service policy includes other rules you want to maintain, simply remove the redirection command from the relevant policy map, or the entire traffic class if redirection is the only action for the class. You can remove the policies using CLI or ASDM.
- Obtain both the ASA FirePOWER Boot Image and System Software packages from Cisco.com.

Procedure

Step 1 Download the boot image to the ASA. Do not transfer the system software; it is downloaded later to the SSD. You have the following options:

- ASDM—First, download the boot image to your workstation, or place it on an FTP, TFTP, HTTP, HTTPS, SMB, or SCP server. Then, in ASDM, choose **Tools > File Management**, and then choose the appropriate **File Transfer** command, either **Between Local PC and Flash** or **Between Remote Server and Flash**. Transfer the boot software to disk0 on the ASA.
- ASA CLI—First, place the boot image on a TFTP, FTP, HTTP, or HTTPS server, then use the **copy** command to download it to flash. The following example uses TFTP:

  ```
  ciscoasa# copy tftp://10.1.1.89/asasfr-5500x-boot-5.4.1-58.img
  disk0:/asasfr-5500x-boot-5.4.1-58.img
  ```

Step 2 Download the ASA FirePOWER system software from Cisco.com to an HTTP, HTTPS, or FTP server accessible from the ASA FirePOWER management interface. Do not download it to disk0 on the ASA.

Step 3 Set the ASA FirePOWER module boot image location in ASA disk0 by entering the following command:

```
hostname# sw-module module sfr recover configure image disk0:file_path
```

Example:

```
hostname# sw-module module sfr recover configure image
disk0:asasfr-5500x-boot-5.3.1-58.img
```

If you see a message such as "ERROR: Another service (cxsc) is running, only one service is allowed to run at any time," it means that you already have a different software module configured. You must shut it down and remove it to install a new module as described in the prerequisites section above.

Step 4 Load the ASA FirePOWER boot image:

```
hostname# sw-module module sfr recover boot
```

Step 5 Wait approximately 5-15 minutes for the ASA FirePOWER module to boot up, and then open a console session to the now-running ASA FirePOWER boot image. You might need to press enter after opening the session to get to the login prompt. The default username is **admin** and the default password is **Admin123**.

```
hostname# session sfr console
Opening console session with module sfr.
Connected to module sfr. Escape character sequence is 'CTRL-^X'.

Cisco ASA SFR Boot Image 5.3.1
asasfr login: admin
Password: Admin123
```

If the module boot has not completed, the **session** command will fail with a message about not being able to connect over ttyS1. Wait and try again.

Step 6 Configure the system so that you can install the system software package:

```
asasfr-boot> setup


               Welcome to SFR Setup
                [hit Ctrl-C to abort]
              Default values are inside []
```

You are prompted for the following. Note that the management address and gateway, and DNS information, are the key settings to configure.

- Host name—Up to 65 alphanumeric characters, no spaces. Hyphens are allowed.
- Network address—You can set static IPv4 or IPv6 addresses, or use DHCP (for IPv4) or IPv6 stateless autoconfiguration.
- DNS information—You must identify at least one DNS server, and you can also set the domain name and search domain.
- NTP information—You can enable NTP and configure the NTP servers, for setting system time.

Step 7 Install the System Software image:

```
asasfr-boot> system install [noconfirm] url
```

Include the **noconfirm** option if you do not want to respond to confirmation messages. Use an HTTP, HTTPS, or FTP URL; if a username and password are required, you will be prompted to supply them.

When installation is complete, the system reboots. Allow 10 or more minutes for application component installation and for the ASA FirePOWER services to start. (The **show module sfr** output should show all processes as Up.)

For example:

```
asasfr-boot> system install http://upgrades.example.com/packages/asasfr-sys-5.3.1-44.pkg
Verifying
Downloading
Extracting
Package Detail
        Description:                    Cisco ASA-FirePOWER 5.3.1-44 System Install
        Requires reboot:                Yes

Do you want to continue with upgrade? [y]: y
Warning: Please do not interrupt the process or turn off the system.
Doing so might leave system in unusable state.

Upgrading
Starting upgrade process ...
Populating new system image

Reboot is required to complete the upgrade. Press 'Enter' to reboot the system.
(press Enter)
Broadcast message from root (ttyS1) (Mon Feb 17 19:28:38 2014):

The system is going down for reboot NOW!
Console session with module sfr terminated.
```

Step 8 Open a session to the ASA FirePOWER module. You will see a different login prompt because you are logging into the fully functional module.

```
hostname# session sfr console
Opening console session with module sfr.
Connected to module sfr. Escape character sequence is 'CTRL-^X'.

Sourcefire ASA5555 v5.4.1 (build 58)
Sourcefire3D login:
```

Step 9 See Configure ASA FirePOWER Basic Settings, page 7-9 to complete the setup.

Reimage the ASA 5585-X ASA FirePOWER Hardware Module

If you need to reimage the ASA FirePOWER hardware module in an ASA 5585-X for any reason, you need to install both the Boot Image and a System Software package, in that order. You must install both packages to have a functioning system. Under normal circumstances, you do not need to reimage the system to install upgrade packages.

To install the boot image, you need to TFTP boot the image from the Management-0 port on the ASA FirePOWER SSP by logging into the module's Console port. Because the Management-0 port is on an SSP in the first slot, it is also known as Management1/0, but ROMMON recognizes it as Management-0 or Management0/1.

Before You Begin

To accomplish a TFTP boot, you must:

- Place the Boot Image and a System Software package on a TFTP server that can be accessed through the Management1/0 interface on the ASA FirePOWER module.

- Connect Management1/0 to the network. You must use this interface to TFTP boot the Boot Image.

Procedure

Step 1 Connect to the module console port.

Step 2 Reload the system:

```
system reboot
```

Step 3 When prompted, break out of the boot by pressing **Esc**. If you see grub start to boot the system, you have waited too long.

This will place you at the ROMMON prompt.

Step 4 At the ROMMON prompt, enter **set** and configure the following parameters:

- ADDRESS—The management IP address of the module.
- SERVER—The IP address of the TFTP server.
- GATEWAY—The gateway address to the TFTP server. If the TFTP server and management address are on the same subnet, do not configure the gateway or TFTP boot will fail.
- IMAGE—The Boot Image path and image name on the TFTP server. For example, if you place the file on the TFTP server in /tftpboot/images/filename.img, the IMAGE value is images/filename.img.

For example:

```
ADDRESS=10.5.190.199
SERVER=10.5.11.170
GATEWAY=10.5.1.1
IMAGE=asasfr-boot-5.3.1-26-54.img
```

Step 5 Save the settings:

```
sync
```

Step 6 Initiate the download and boot process:

```
tftp
```

You will see ! marks to indicate progress. When the boot completes after several minutes, you will see a login prompt.

Step 7 Log in as **admin**, with the password **Admin123**.

Step 8 Configure the system so that you can install the system software package:

```
setup
```

You are prompted for the following. Note that the management address and gateway, and DNS information, are the key settings to configure.

- Host name—Up to 65 alphanumeric characters, no spaces. Hyphens are allowed.
- Network address—You can set static IPv4 or IPv6 addresses, or use DHCP (for IPv4) or IPv6 stateless autoconfiguration.
- DNS information—You must identify at least one DNS server, and you can also set the domain name and search domain.
- NTP information—You can enable NTP and configure the NTP servers, for setting system time.

Step 9 Install the System Software image:

```
system install [noconfirm] url
```

Example:

```
asasfr-boot> system install http://upgrades.example.com/packages/asasfr-sys-5.4.1-54.pkg
```

Include the **noconfirm** option if you do not want to respond to confirmation messages.

When installation is complete, the system reboots. Allow 10 or more minutes for application component installation and for the ASA FirePOWER services to start.

Step 10 When the boot completes, log in as **admin** with the password **Sourcefire**.

Step 11 See Configure ASA FirePOWER Basic Settings, page 7-9 to complete the setup.

Reset the Password

If you forget the password for the admin user, another user with CLI Configuration permissions can log in and change the password.

If there are no other users with the required permissions, you can reset the admin password from the ASA.

Before You Begin

- In multiple context mode, perform this procedure in the system execution space.
- The **password-reset** option on the ASA **hw-module** and **sw-module** commands does not work with ASA FirePOWER.

Procedure

Step 1 Reset the module password for the user **admin** to the default, **Sourcefire**:

```
session {1 | sfr} do password-reset
```

Use **1** for a hardware module, **sfr** for a software module.

Reload or Reset the Module

You can reload, or to reset and then reload, the module from the ASA.

Before You Begin

In multiple context mode, perform this procedure in the system execution space.

Procedure

Step 1 Enter one of the following commands:

- Hardware module (ASA 5585-X):

  ```
  hw-module module 1 {reload | reset}
  ```

- Software module (all other models):

```
sw-module module sfr {reload | reset}
```

Shut Down the Module

Shutting down the module software prepares the module to be safely powered off without losing configuration data.

Before You Begin

- In multiple context mode, perform this procedure in the system execution space.
- If you reload the ASA, the module is not automatically shut down, so we recommend shutting down the module before reloading the ASA.

Procedure

Step 1 Enter one of the following commands:

- Hardware module (ASA 5585-X):

  ```
  hw-module module 1 shutdown
  ```

- Software module (all other models):

  ```
  sw-module module sfr shutdown
  ```

Uninstall a Software Module Image

You can uninstall a software module image and its associated configuration.

Before You Begin

In multiple context mode, perform this procedure in the system execution space.

Procedure

Step 1 Uninstall the software module image and associated configuration:

```
sw-module module sfr uninstall

Module sfr will be uninstalled. This will completely remove the disk image
associated with the sw-module including any configuration that existed within it.

Uninstall module sfr? [confirm]
```

Step 2 Reload the ASA:

```
reload
```

You must reload the ASA before you can install a new module.

Session to the Software Module From the ASA

Use the ASA FirePOWER CLI to configure basic network settings and to troubleshoot the module.

To access the ASA FirePOWER software module CLI from the ASA, you can session from the ASA. (You cannot session to a hardware module running on a 5585-X.)

You can either session to the module (using Telnet) or create a virtual console session. A console session might be useful if the control plane is down and you cannot establish a Telnet session. In multiple context mode, session from the system execution space.

In either a Telnet or a Console session, you are prompted for a username and password. You can log in with any username configured on the ASA FirePOWER. Initially, the **admin** username is the only one configured (and it is always available). The initial default password is **Sourcefire** for the full image, and **Admin123** for the boot image.

- Telnet session:

  ```
  session sfr
  ```

  When in the ASA FirePOWER CLI, to exit back to the ASA CLI, enter any command that would log you out of the module, such as **logout** or **exit**, or press **Ctrl-Shift-6, x**.

- Console session:

  ```
  session sfr console
  ```

  The only way out of a console session is to press **Ctrl-Shift-6, x**. Logging out of the module leaves you at the module login prompt.

Note Do not use the **session sfr console** command in conjunction with a terminal server where **Ctrl-Shift-6, x** is the escape sequence to return to the terminal server prompt. **Ctrl-Shift-6, x** is also the sequence to escape the ASA FirePOWER console and return to the ASA prompt. Therefore, if you try to exit the ASA FirePOWER console in this situation, you instead exit all the way to the terminal server prompt. If you reconnect the terminal server to the ASA, the ASA FirePOWER console session is still active; you can never exit to the ASA prompt. You must use a direct serial connection to return the console to the ASA prompt. Use the **session sfr** command instead of the console command when facing this situation.

Upgrade the System Software

Before applying an upgrade, ensure that the ASA is running the minimum required release for the new version; you might need to upgrade the ASA prior to upgrading the module. For more information about applying upgrades, see the *FireSIGHT System User Guide* or the online help in FireSIGHT Management Center.

For ASDM management, you can apply upgrades to the system software and components using **Configuration > ASA FirePOWER Configuration > Updates**. Click **Help** on the Updates page for more information.

Monitoring the ASA FirePOWER Module

The following topics provide guidance on monitoring the module. For ASA FirePOWER-related syslog messages, see the syslog messages guide. ASA FirePOWER syslog messages start with message number 434001.

Showing Module Status

To check the status of a module, enter one of the following commands:

- **show module** [**1** | **sfr**] [**details**]

  Shows the status of modules. Include the 1 (for hardware modules) or sfr (for software modules) keyword to see status specific to the ASA FirePOWER module. Include the details keyword to get additional information, including the address of the device that manages the module.

- **show module sfr recover**

  Displays the location of the boot image used when installing the module.

The following is sample output from the **show module** command for an ASA 5585-X with an ASA FirePOWER hardware module installed:

```
hostname# show module
Mod  Card Type                                    Model              Serial No.
---- -------------------------------------------- ------------------ -----------
   0 ASA 5585-X Security Services Processor-10 wi ASA5585-SSP-10     JAF1507AMKE
   1 ASA 5585-X FirePOWER Security Services Proce ASA5585-SSP-SFR10  JAF1510BLSA

Mod  MAC Address Range                 Hw Version   Fw Version   Sw Version
---- --------------------------------- ------------ ------------ ---------------
   0 5475.d05b.1100 to 5475.d05b.110b  1.0          2.0(7)0      100.10(0)8
   1 5475.d05b.2450 to 5475.d05b.245b  1.0          2.0(13)0     5.3.1-44

Mod  SSM Application Name           Status           SSM Application Version
---- ------------------------------ ---------------- --------------------------
   1 FirePOWER                      Up               5.3.1-44

Mod  Status             Data Plane Status     Compatibility
---- ------------------ --------------------- -------------
   0 Up Sys             Not Applicable
   1 Up                 Up
```

The following example shows the details for a software module. Note that DC Addr indicates the address of the FireSIGHT Management Center that manages this device.

```
hostname# show module sfr details
Getting details from the Service Module, please wait...

Card Type:          FirePOWER Services Software Module
Model:              ASA5555
Hardware version:   N/A
Serial Number:      FCH1714J6HP
Firmware version:   N/A
Software version:   5.3.1-100
MAC Address Range:  bc16.6520.1dcb to bc16.6520.1dcb
```

```
App. name:           ASA FirePOWER
App. Status:         Up
App. Status Desc:    Normal Operation
App. version:        5.3.1-100
Data Plane Status:   Up
Status:              Up
DC addr:             10.89.133.202
Mgmt IP addr:        10.86.118.7
Mgmt Network mask:   255.255.252.0
Mgmt Gateway:        10.86.116.1
Mgmt web ports:      443
Mgmt TLS enabled:    true
```

The following example shows the location of the ASA FirePOWER boot image that was used with the **sw-module module sfr recover** command when installing the module.

```
hostname# show module sfr recover
Module sfr recover parameters...
Boot Recovery Image: No
Image File Path:     disk0:/asasfr-5500x-boot-5.3.1-44.img
```

Showing Module Statistics

Use the **show service-policy sfr** command to display statistics and status for each service policy that includes the **sfr** command. Use **clear service-policy** to clear the counters.

The following example shows the ASA FirePOWER service policy and the current statistics as well as the module status. In monitor-only mode, the input counters remain at zero.

```
ciscoasa# show service-policy sfr

Global policy:
  Service-policy: global_policy
    Class-map: my-sfr-class
      SFR: card status Up, mode fail-close
        packet input 2626422041, packet output 2626877967, drop 0, reset-drop 0, proxied 0
```

Monitoring Module Connections

To show connections through the ASA FirePOWER module, enter one of the following commands:

- **show asp table classify domain sfr**

  Shows the NP rules created to send traffic to the ASA FirePOWER module.

- **show asp drop**

  Shows dropped packets. The drop types are explained below.

- **show conn**

  Shows if a connection is being forwarded to a module by displaying the ‘X - inspected by service module’ flag.

The **show asp drop** command can include the following drop reasons related to the ASA FirePOWER module.

Frame Drops:

- sfr-bad-tlv-received—This occurs when ASA receives a packet from FirePOWER without a Policy ID TLV. This TLV must be present in non-control packets if it does not have the Standby/Active bit set in the actions field.
- sfr-request—The frame was requested to be dropped by FirePOWER due a policy on FirePOWER whereby FirePOWER would set the actions to Deny Source, Deny Destination, or Deny Pkt. If the frame should not have been dropped, review the policies on the module that are denying the flow.
- sfr-fail-close—The packet is dropped because the card is not up and the policy configured was ‘fail-close’ (rather than ‘fail-open’ which allows packets through even if the card was down). Check card status and attempt to restart services or reboot it.
- sfr-fail—The FirePOWER configuration was removed for an existing flow and we are not able to process it through FirePOWER it will be dropped. This should be very unlikely.
- sfr-malformed-packet—The packet from FirePOWER contains an invalid header. For instance, the header length may not be correct.
- sfr-ha-request—This counter is incremented when the security appliance receives a FirePOWER HA request packet, but could not process it and the packet is dropped.
- sfr-invalid-encap—This counter is incremented when the security appliance receives a FirePOWER packet with invalid message header, and the packet is dropped.
- sfr-bad-handle-received—Received Bad flow handle in a packet from FirePOWER Module, thus dropping flow. This counter is incremented, flow and packet are dropped on ASA as the handle for FirePOWER flow has changed in flow duration.
- sfr-rx-monitor-only—This counter is incremented when the security appliance receives a FirePOWER packet when in monitor-only mode, and the packet is dropped.

Flow Drops:

- sfr-request—The FirePOWER requested to terminate the flow. The actions bit 0 is set.
- reset-by-sfr—The FirePOWER requested to terminate and reset the flow. The actions bit 1 is set.
- sfr-fail-close—The flow was terminated because the card is down and the configured policy was 'fail-close'.

Examples for the ASA FirePOWER Module

The following example diverts all HTTP traffic to the ASA FirePOWER module, and blocks all HTTP traffic if the module fails for any reason:

```
hostname(config)# access-list ASASFR permit tcp any any eq 80
hostname(config)# class-map my-sfr-class
hostname(config-cmap)# match access-list ASASFR
hostname(config-cmap)# policy-map my-sfr-policy
hostname(config-pmap)# class my-sfr-class
hostname(config-pmap-c)# sfr fail-close
hostname(config-pmap-c)# service-policy my-sfr-policy global
```

The following example diverts all IP traffic destined for the 10.1.1.0 network and the 10.2.1.0 network to the ASA FirePOWER module, and allows all traffic through if the module fails for any reason.

```
hostname(config)# access-list my-sfr-acl permit ip any 10.1.1.0 255.255.255.0
hostname(config)# access-list my-sfr-acl2 permit ip any 10.2.1.0 255.255.255.0
hostname(config)# class-map my-sfr-class
hostname(config-cmap)# match access-list my-sfr-acl
hostname(config)# class-map my-sfr-class2
hostname(config-cmap)# match access-list my-sfr-acl2
hostname(config-cmap)# policy-map my-sfr-policy
hostname(config-pmap)# class my-sfr-class
hostname(config-pmap-c)# sfr fail-open
hostname(config-pmap)# class my-sfr-class2
hostname(config-pmap-c)# sfr fail-open
hostname(config-pmap-c)# service-policy my-sfr-policy interface outside
```

History for the ASA FirePOWER Module

Feature	Platform Releases	Description
ASA 5585-X (all models) support for the matching ASA FirePOWER SSP hardware module. ASA 5512-X through ASA 5555-X support for the ASA FirePOWER software module.	ASA 9.2(2.4) ASA FirePOWER 5.3.1	The ASA FirePOWER module supplies next-generation firewall services, including Next-Generation IPS (NGIPS), Application Visibility and Control (AVC), URL filtering, and Advanced Malware Protection (AMP).You can use the module in single or multiple context mode, and in routed or transparent mode. We introduced or modified the following commands: **capture interface asa_dataplane**, **debug sfr**, **hw-module module 1 reload**, **hw-module module 1 reset**, **hw-module module 1 shutdown**, **session do setup host ip, session do get-config, session do password-reset, session sfr, sfr, show asp table classify domain sfr**, **show capture**, **show conn**, **show module sfr**, **show service-policy, sw-module sfr**.
ASA 5506-X support for the ASA FirePOWER software module, including support for configuring the module in ASDM	ASA 9.3(2) ASDM 7.3(3) ASA FirePOWER 5.4.1	You can run the ASA FirePOWER software module on the ASA 5506-X. You can manage the module using FireSIGHT Management Center, or you can use ASDM.
ASA FirePOWER passive monitor-only mode using traffic redirection interfaces	ASA 9.3(2) ASA FirePOWER 5.4.1	You can now configure a traffic forwarding interface to send traffic to the module instead of using a service policy. In this mode, neither the module nor the ASA affects the traffic. We fully supported the following command: **traffic-forward sfr monitor-only**. You can configure this in CLI only.

Feature	Platform Releases	Description
ASA 5506W-X, ASA 5506H-X, ASA 5508-X, and ASA 5516-X support for the ASA FirePOWER software module, including support for configuring the module in ASDM	ASA 9.4(1) ASDM 7.4(1) ASA FirePOWER 5.4.1	You can run the ASA FirePOWER software module on the ASA 5506W-X, ASA 5506H-X, ASA 5508-X, and ASA 5516-X. You can manage the module using FireSIGHT Management Center or you can use ASDM.

ASA and Cisco Cloud Web Security

Cisco Cloud Web Security (also known as ScanSafe) provides web security and web filtering services through the Software-as-a-Service (SaaS) model. Enterprises with the ASA in their network can use Cloud Web Security services without having to install additional hardware.

Information About Cisco Cloud Web Security

When you enable Cloud Web Security on the ASA, the ASA transparently redirects selected HTTP and HTTPS traffic to the Cloud Web Security proxy servers based on service policy rules. The Cloud Web Security proxy servers then scan the content and allow, block, or send a warning about the traffic based on the policy configured in Cisco ScanCenter to enforce acceptable use and to protect users from malware.

The ASA can optionally authenticate and identify users with Identity Firewall and AAA rules. The ASA encrypts and includes the user credentials (including usernames and user groups) in the traffic it redirects to Cloud Web Security. The Cloud Web Security service then uses the user credentials to match the traffic to the policy. It also uses these credentials for user-based reporting. Without user authentication, the ASA can supply an (optional) default username and group, although usernames and groups are not required for the Cloud Web Security service to apply policy.

You can customize the traffic you want to send to Cloud Web Security when you create your service policy rules. You can also configure a “whitelist” so that a subset of web traffic that matches the service policy rule instead goes directly to the originally requested web server and is not scanned by Cloud Web Security.

You can configure a primary and a backup Cloud Web Security proxy server, each of which the ASA polls regularly to check for availability.

User Identity and Cloud Web Security

You can use user identity to apply policy in Cloud Web Security. User identity is also useful for Cloud Web Security reporting. User identity is not required to use Cloud Web Security. There are other methods to identify traffic for Cloud Web Security policy.

You can use the following methods of determining the identity of a user or of providing a default identity:

- Identity firewall—When the ASA uses identity firewall with Active Directory (AD), the username and group is retrieved from the AD agent. Users and groups are retrieved when you use them in an ACL in a feature such as an access rule or in your service policy, or by configuring the user identity monitor to download user identity information directly.

  For information about configuring IDFW, see the general operations configuration guide.
- AAA rules—When the ASA performs user authentication using a AAA rule, the username is retrieved from the AAA server or local database. Identity from AAA rules does not include group information. If you configure a default group, these users are associated with that default group. For information about configuring AAA rules, see the legacy feature guide.
- Default username and group—For traffic that does not have an associated user name or group, you can configure an optional default username and group name. These defaults are applied to all users that match a service policy rule for Cloud Web Security.

Authentication Keys

Each ASA must use an authentication key that you obtain from Cloud Web Security. The authentication key lets Cloud Web Security identify the company associated with web requests and ensures that the ASA is associated with a valid customer.

You can use one of two types of authentication keys for your ASA: the company key or the group key.

- **Company authentication key**—You can use a company authentication key on multiple ASAs within the same company. This key simply enables the Cloud Web Security service for your ASAs.
- **Group authentication key**—A Group authentication key is a special key unique to each ASA that performs two functions:
  - Enables the Cloud Web Security service for one ASA.
  - Identifies all traffic from the ASA so you can create ScanCenter policy per ASA.

You generate these keys in ScanCenter (https://scancenter.scansafe.com/portal/admin/login.jsp). For more information, see the Cloud Web Security documentation:

http://www.cisco.com/c/en/us/support/security/cloud-web-security/products-installation-and-configuration-guides-list.html

ScanCenter Policy

In ScanCenter, traffic is matched against policy rules in order until a rule is matched. Cloud Web Security then applies the configured action for the rule, allowing or blocking the traffic, or warning the user. With warnings, the user has the option to continue on to the web site.

You configure the URL filtering policies in ScanCenter, not in the ASA.

However, part of the policy is to whom the policy applies. User traffic can match a policy rule in ScanCenter based on group association: a *directory group* or a *custom group*. Group information is included in the requests redirected from the ASA, so you need to understand what group information you might get from the ASA.

- Directory Groups, page 8-3
- Custom Groups, page 8-3
- How Groups and the Authentication Key Interoperate, page 8-4

Directory Groups

Directory groups define the group to which traffic belongs. When using the identity firewall, the group, if present, is included in the client's HTTP request. If you do not use identity firewall, you can configure a default group for traffic matching an ASA rule for Cloud Web Security inspection.

In ScanCenter, when you configure a directory group in a policy, you must enter the group name exactly.

- Identity firewall group names are sent in the following format.

  *domain-name**group-name*

  Note that on the ASA, the format is *domain-name**group-name*. However, the ASA modifies the name to use only one backslash (\) to conform to typical ScanCenter notation when including the group in the redirected HTTP request.

- The default group name is sent in the following format:

  [*domain*\]*group-name*

  On the ASA, you need to configure the optional domain name to be followed by 2 backslashes (\\); however, the ASA modifies the name to use only one backslash (\) to conform to typical ScanCenter notation. For example, if you specify "Cisco\\Boulder1," the ASA modifies the group name to be "Cisco\Boulder1" with only one backslash (\) when sending the group name to Cloud Web Security.

Custom Groups

Custom groups are defined using one or more of the following criteria:

- ScanCenter Group authentication key—You can generate a Group authentication key for a custom group. Then, if you identify this group key when you configure the ASA, all traffic from the ASA is tagged with the Group key.
- Source IP address—You can identify source IP addresses in the custom group. Note that the ASA service policy is based on source IP address, so you might want to configure any IP address-based policy on the ASA instead.
- Username—You can identify usernames in the custom group.
  - Identity firewall usernames are sent in the following format:

    *domain-name**username*
  - AAA usernames, when using RADIUS or TACACS+, are sent in the following format:

    LOCAL*username*
  - AAA usernames, when using LDAP, are sent in the following format:

    *domain-name**username*

– For the default username, it is sent in the following format:

  [*domain-name*\]*username*

  For example, if you configure the default username to be “Guest,” then the ASA sends “Guest.” If you configure the default username to be “Cisco\Guest,” then the ASA sends “Cisco\Guest.”

How Groups and the Authentication Key Interoperate

Unless you need the per-ASA policy that a custom group plus group key provides, you will likely use a company key. Note that not all custom groups are associated with a group key. You can use non-keyed custom groups to identify IP addresses or usernames, and use them in your policy along with rules that use directory groups.

Even if you do want per-ASA policy and are using a group key, you can also use the matching capability provided by directory groups and non-keyed custom groups. In this case, you might want an ASA-based policy, with some exceptions based on group membership, IP address, or username. For example, if you want to exempt users in the America\Management group across all ASAs:

1. Add a directory group for America\Management.
2. Add an exempt rule for this group.
3. Add rules for each custom group plus group key after the exempt rule to apply policy per-ASA.
4. Traffic from users in America\Management will match the exempt rule, while all other traffic will match the rule for the ASA from which it originated.

Many combinations of keys, groups, and policy rules are possible.

Failover from Primary to Backup Proxy Server

When you subscribe to the Cisco Cloud Web Security service, you are assigned a primary Cloud Web Security proxy server and backup proxy server.

If any client is unable to reach the primary server, then the ASA starts polling the tower to determine availability. (If there is no client activity, the ASA polls every 15 minutes.) If the proxy server is unavailable after a configured number of retries (the default is 5; this setting is configurable), the server is declared unreachable, and the backup proxy server becomes active.

After a failover to the backup server, the ASA continues to poll the primary server. If the primary server becomes reachable, then the ASA returns to using the primary server.

You can also choose how the ASA handles web traffic when it cannot reach either the primary or backup Cloud Web Security proxy server. It can block or allow all web traffic. By default, it blocks web traffic.

Licensing Requirements for Cisco Cloud Web Security

Model	License Requirement
ASAv	Standard or Premium License.
All other models	Strong Encryption (3DES/AES) License to encrypt traffic between the ASA and the Cloud Web Security server.

On the Cloud Web Security side, you must purchase a Cisco Cloud Web Security license and identify the number of users that the ASA handles. Then log into ScanCenter and generate your authentication keys.

Guidelines for Cloud Web Security

Context Mode Guidelines

Supported in single and multiple context modes.

In multiple context mode, the server configuration is allowed only in the system context, and the service policy rule configuration is allowed only in the security contexts.

Each context can have its own authentication key, if desired.

Firewall Mode Guidelines

Supported in routed firewall mode only. Does not support transparent firewall mode.

IPv6 Guidelines

Does not support IPv6. Cloud Web Security currently supports only IPv4 addresses. If you use IPv6 internally, use NAT 64 to translate IPv6 addresses to IPv4 for any IPv6 flows that need to be sent to Cloud Web Security.

Additional Guidelines

- Cloud Web Security is not supported with ASA clustering.
- You cannot use Cloud Web Security on the same traffic you redirect to a module that can also perform URL filtering, such as ASA CX and ASA FirePOWER. The traffic is sent to the modules only, not to the Cloud Web Security servers.
- Clientless SSL VPN is not supported with Cloud Web Security; be sure to exempt any clientless SSL VPN traffic from the ASA service policy for Cloud Web Security.
- When an interface to the Cloud Web Security proxy servers goes down, output from the **show scansafe server** command shows both servers up for approximately 15-25 minutes. This condition may occur because the polling mechanism is based on the active connection, and because that interface is down, it shows zero connection, and it takes the longest poll time approach.
- Cloud Web Security inspection is compatible with HTTP inspection for the same traffic.
- Cloud Web Security is not supported with extended PAT or any application that can potentially use the same source port and IP address for separate connections. For example, if two different connections (targeted to separate servers) use extended PAT, the ASA might reuse the same source IP and source port for both connection translations because they are differentiated by the separate destinations. When the ASA redirects these connections to the Cloud Web Security server, it replaces the destination with the Cloud Web Security server IP address and port (8080 by default). As a result, both connections now appear to belong to the same flow (same source IP/port and destination IP/port), and return traffic cannot be untranslated properly.
- The default inspection traffic class does not include the default ports for the Cloud Web Security inspection (80 and 443).

Configure Cisco Cloud Web Security

Before you configure Cloud Web Security, obtain a license and the addresses of the proxy servers you will use. Also, generate your authentication keys. Learn more about at Cloud Web Security http://www.cisco.com/go/cloudwebsecurity.

Use the following process to configure the ASA to redirect web traffic to Cloud Web Security.

Before You Begin

If you want to send user identity information to Cloud Web Security, configure one of the following on the ASA:

- Identity firewall (username and group).
- AAA rules (username only)—See the legacy feature guide.

If you want to use fully-qualified domain names (FQDN), such as www.example.com, you must configure a DNS server for the ASA.

Procedure

Step 1 Configure Communications with the Cloud Web Security Proxy Server, page 8-6.

Step 2 (Optional.) Identify Whitelisted Traffic, page 8-8.

Step 3 Configure a Service Policy to Send Traffic to Cloud Web Security, page 8-9.

Step 4 (Optional.) Configure the User Identity Monitor, page 8-13

Step 5 Configure the Cloud Web Security Policy, page 8-14.

Configure Communications with the Cloud Web Security Proxy Server

You must identify the Cloud Web Security proxy servers so that user web requests can be redirected properly.

In multiple context mode, you must configure the proxy servers in the system context, then enable Cloud Web Security per context. Thus, you can use the service in some contexts but not in others.

Before You Begin

- You must configure a DNS server for the ASA to use fully-qualified domain names for the proxy servers.
- (Multiple context mode.) You must configure a route pointing to the Cloud Web Security proxy servers in both the system context and the specific contexts. This ensures that the Cloud Web Security proxy servers do not become unreachable in the Active/Active failover scenario.

Procedure

Step 1 Enter ScanSafe general-options configuration mode. In multiple context mode, do this in the system context.

```
scansafe general-options
```

Example

```
hostname(config)# scansafe general-options
```

Step 2 Configure the primary and secondary Cloud Web Security proxy servers.

```
server primary {ip ip_address | fqdn fqdn} [port port]
server backup {ip ip_address | fqdn fqdn} [port port]
```

Example

```
hostname(cfg-scansafe)# server primary ip 192.168.43.10
hostname(cfg-scansafe)# server backup fqdn server.example.com
```

When you subscribe to the Cisco Cloud Web Security service, you are assigned primary and backup Cloud Web Security proxy servers. Enter their IP addresses (**ip**), or fully-qualified domain names (**fqdn**), on these commands.

By default, the Cloud Web Security proxy server uses port 8080 for both HTTP and HTTPS traffic; do not change this value unless directed to do so.

Step 3 (Optional.) Configure the number of consecutive polling failures to the Cloud Web Security proxy server before determining the server is unreachable.

```
retry-count value
```

Example

```
hostname(cfg-scansafe)# retry-count 2
```

Polls are performed every 30 seconds. Valid values are from 2 to 100, and the default is 5.

Step 4 Configure the authentication key that the ASA sends to the Cloud Web Security proxy servers to indicate from which organization the request comes.

```
license hex_key
```

Example

```
hostname(cfg-scansafe)# license F12A588FE5A0A4AE86C10D222FC658F3
```

The authentication key is a 16-byte hexidecimal number. It can be a company or group key.

Step 5 (Multiple context mode only.) Switch to each context where you want to use the service and enable it.

```
scansafe [license hex_key]
```

Example

```
hostname(config)# changeto context one
hostname/one(config)# scansafe
```

You can optionally enter a separate authentication key for each context. If you do not include an authentication key, the one configured for the system context is used.

Examples

The following example configures a primary and backup server:

```
scansafe general-options
 server primary ip 10.24.0.62 port 8080
 server backup ip 10.10.0.7 port 8080
 retry-count 7
 license 366C1D3F5CE67D33D3E9ACEC265261E5
```

The following sample configuration enables Cloud Web Security in context one with the default license and in context two with the license key override:

```
! System Context
!
scansafe general-options
server primary ip 180.24.0.62 port 8080
license 366C1D3F5CE67D33D3E9ACEC265261E5
!
context one
 allocate-interface GigabitEthernet0/0.1
 allocate-interface GigabitEthernet0/1.1
 allocate-interface GigabitEthernet0/3.1
 scansafe
 config-url disk0:/one_ctx.cfg
!
context two
 allocate-interface GigabitEthernet0/0.2
 allocate-interface GigabitEthernet0/1.2
 allocate-interface GigabitEthernet0/3.2
 scansafe license 366C1D3F5CE67D33D3E9ACEC26789534
 config-url disk0:/two_ctx.cfg
!
```

Identify Whitelisted Traffic

If you use identity firewall or AAA rules, you can configure the ASA so that web traffic from specific users or groups that otherwise match the service policy rule is not redirected to the Cloud Web Security proxy server for scanning. This process is called “whitelisting” traffic.

You configure the whitelist in a ScanSafe inspection class map. You can use usernames and group names derived from both identity firewall and AAA rules. You cannot whitelist based on IP address or on destination URL.

When you configure your Cloud Web Security service policy rule, you refer to the class map in your policy. Although you can achieve the same results of exempting traffic based on user or group when you configure the traffic matching criteria (with ACLs) in the service policy rule, you might find it more straightforward to use a whitelist instead.

Procedure

Step 1 Create the class map.

```
hostname(config)# class-map type inspect scansafe [match-all | match-any] class_map_name
hostname(config-cmap)#
```

Where the *class_map_name* is the name of the class map. The **match-all** keyword is the default, and specifies that traffic must match all criteria to match the class map. The match-any keyword specifies that the traffic matches the class map if it matches at least one **match** statement. The CLI enters class-map configuration mode, where you can enter one or more **match** commands.

Example

```
hostname(config)# class-map type inspect scansafe match-any whitelist1
```

Step 2 Specify the whitelisted users and groups.

```
match [not] {[user username] [group groupname]}
```

The **match** keyword specifies a user or group to whitelist, or both.

The **match not** keyword specifies that the user or group should be filtered using Cloud Web Security. For example, if you whitelist the group "cisco," but you want to scan traffic from users "johncrichton" and "aerynsun," which are members of that group, you can specify **match not** for those users. Repeat this command to add as many users and groups as needed.

Example

The following example whitelists the same users and groups for the HTTP and HTTPS inspection policy maps:

```
hostname(config)# class-map type inspect scansafe match-any whitelist1
hostname(config-cmap)# match user user1 group cisco
hostname(config-cmap)# match user user2
hostname(config-cmap)# match group group1
hostname(config-cmap)# match user user3 group group3

hostname(config)# policy-map type inspect scansafe cws_inspect_pmap1
hostname(config-pmap)# parameters
hostname(config-pmap-p)# http
hostname(config-pmap-p)# default group default_group
hostname(config-pmap-p)# class whitelist1
hostname(config-pmap-c)# whitelist

hostname(config)# policy-map type inspect scansafe cws_inspect_pmap2
hostname(config-pmap)# parameters
hostname(config-pmap-p)# https
hostname(config-pmap-p)# default group2 default_group2
hostname(config-pmap-p)# class whitelist1
hostname(config-pmap-c)# whitelist
```

Configure a Service Policy to Send Traffic to Cloud Web Security

Your service policy consists of multiple service policy rules, applied globally, or applied to each interface. Each service policy rule can either send traffic to Cloud Web Security (Match) or exempt traffic from Cloud Web Security (Do Not Match).

Create rules for traffic destined for the Internet. The order of these rules is important. When the ASA decides whether to forward or exempt a packet, the ASA tests the packet with each rule in the order in which the rules are listed. After a match is found, no more rules are checked. For example, if you create a rule at the beginning of a policy that explicitly Matches all traffic, no further statements are ever checked.

Before You Begin

If you need to use a whitelist to exempt some traffic from being sent to Cloud Web Security, first create the whitelist so you can refer to it in your service policy rule.

Procedure

Step 1 Create the ScanSafe inspection policy maps. You need to define separate maps for HTTP and HTTPS.

a. Create the ScanSafe inspection policy map.

```
hostname(config)# policy-map type inspect scansafe policy_map_name
```

```
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

b. Enter parameters configuration mode.

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

c. Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- {**http** | **https**}—The service type for this map. You can only specify one service type per map, so you need separate maps for HTTP and HTTPS.
- **default** {[**user** *username*] [**group** *groupname*]}—(Optional.) The default user or group name, or both. If the ASA cannot determine the identity of the user coming into the ASA, then the default user and group is included in the HTTP request sent to Cloud Web Security. You can define policies in ScanCenter for this user or group name.

d. (Optional.) If you defined a whitelist, identify the class and use the **whitelist** command to mark it as a whitelist.

```
hostname(config-pmap-p)# class whitelist1
hostname(config-pmap-c)# whitelist
```

e. Repeat the process to create an inspection policy map for the other protocol, HTTP or HTTPS.

Step 2 Define the classes for the traffic you want to redirect to Cloud Web Security.

ACL matching is the most flexible way to define the class. However, if you want to send all HTTP/HTTPS traffic, you could instead use a port match in the class (**match port tcp 80** and **match port tcp 443**). The following procedure describes an ACL match.

a. Create ACLs (**access-list extended** command) to identify the traffic you want to send to Cloud Web Security. You must create separate ACLs for HTTP and HTTPS traffic. Because Cloud Web Security works on HTTP/HTTPS traffic only, any other traffic defined in the ACL is ignored.

A **permit** ACE sends matching traffic to Cloud Web Security. A **deny** ACE exempts traffic from the service policy rule, so it is not sent to Cloud Web Security. Use **tcp** for the protocol, and identify the port (80 for HTTP, 443 for HTTPS).

When creating your ACLs, consider how you can match appropriate traffic that is destined for the Internet, but not match traffic that is destined for other internal networks. For example, to prevent inside traffic from being sent to Cloud Web Security when the destination is an internal server on the DMZ, be sure to add a deny ACE to the ACL that exempts traffic to the DMZ.

FQDN network objects might be useful in exempting traffic to specific servers. You can also use identity firewall user arguments and Cisco Trustsec security groups to help identify traffic. Note that Trustsec security group information is not sent to Cloud Web Security; you cannot define policy based on security group.

Create as many ACLs as needed for your policy. You can apply redirection to any number of traffic classes.

The following example shows how to exempt HTTP traffic to two servers, but include the remaining traffic. You would create a duplicate ACL for HTTPS traffic, where you simply change the port to 443.

```
hostname(config)# object network cisco1
hostname(config-object-network)# fqdn www.cisco.com

hostname(config)# object network cisco2
```

```
hostname(config-object-network)# fqdn tools.cisco.com

hostname(config)# access-list SCANSAFE_HTTP extended deny tcp any4 object cisco1 eq 80
hostname(config)# access-list SCANSAFE_HTTP extended deny tcp any4 object cisco2 eq 80
hostname(config)# access-list SCANSAFE_HTTP extended permit tcp any4 any4 eq 80
```

b. Create a traffic class for each ACL you defined.

```
hostname(config)# class-map class_name
hostname(config-cmap)# match access-list acl_name
```

Example

```
hostname(config)# class-map cws_class1
hostname(config-cmap)# match access-list SCANSAFE_HTTP
hostname(config)# class-map cws_class2
hostname(config-cmap)# match access-list SCANSAFE_HTTPS
```

Step 3 Create or edit the policy map to redirect the traffic to Cloud Web Security.

a. Add or edit a policy map that sets the actions to take with the class map traffic. In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name. You can only apply one policy to each interface or globally.

```
policy-map name
```

Example:

```
hostname(config)# policy-map global_policy
```

b. Identify one of the traffic class maps you created for Cloud Web Security inspection.

```
class name
```

Example:

```
hostname(config-pmap)# class cws_class1
```

c. Configure ScanSafe inspection for the class.

```
inspect scansafe scansafe_policy_map [fail-open | fail-close]
```

Where:

- *scansafe_policy_map* is the ScanSafe inspection policy map. Ensure that you match the protocols in the class and policy maps (both HTTP or HTTPS).
- Specify **fail-open** to allow traffic to pass through the ASA if the Cloud Web Security servers are unavailable.
- Specify **fail-close** to drop all traffic if the Cloud Web Security servers are unavailable. **fail-close** is the default.

Example:

```
hostname(config-pmap-c)# inspect scansafe cws_inspect_pmap1 fail-open
```

Note If you are editing the default global policy (or any in-use policy) to use a different ScanSafe inspection policy map, you must remove the ScanSafe inspection with the **no inspect scansafe** command, and then re-add it with the new inspection policy map name.

d. Add the class for the other protocol and enable inspection. If you have additional classes, add them also.

```
hostname(config-pmap)# class cws_class2
hostname(config-pmap-c)# inspect scansafe cws_inspect_pmap2 fail-open
```

Step 4 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Examples

The following example configures two classes: one for HTTP and one for HTTPS. Each ACL exempts traffic to www.cisco.com and to tools.cisco.com, and to the DMZ network, for both HTTP and HTTPS. All other traffic is sent to Cloud Web Security, except for traffic from several whitelisted users and groups. The policy is then applied to the inside interface.

```
hostname(config)# class-map type inspect scansafe match-any whitelist1
hostname(config-cmap)# match user user1 group cisco
hostname(config-cmap)# match user user2
hostname(config-cmap)# match group group1
hostname(config-cmap)# match user user3 group group3

hostname(config)# policy-map type inspect scansafe cws_inspect_pmap1
hostname(config-pmap)# parameters
hostname(config-pmap-p)# http
hostname(config-pmap-p)# default group default_group
hostname(config-pmap-p)# class whitelist1
hostname(config-pmap-c)# whitelist

hostname(config)# policy-map type inspect scansafe cws_inspect_pmap2
hostname(config-pmap)# parameters
hostname(config-pmap-p)# https
hostname(config-pmap-p)# default group2 default_group2
hostname(config-pmap-p)# class whitelist1
hostname(config-pmap-c)# whitelist

hostname(config)# object network cisco1
hostname(config-object-network)# fqdn www.cisco.com
hostname(config)# object network cisco2
hostname(config-object-network)# fqdn tools.cisco.com
hostname(config)# object network dmz_network
hostname(config-object-network)# subnet 10.1.1.0 255.255.255.0

hostname(config)# access-list SCANSAFE_HTTP extended deny tcp any4 object cisco1 eq 80
hostname(config)# access-list SCANSAFE_HTTP extended deny tcp any4 object cisco2 eq 80
hostname(config)# access-list SCANSAFE_HTTP extended deny tcp any4 object dmz_network eq
80
hostname(config)# access-list SCANSAFE_HTTP extended permit tcp any4 any4 eq 80

hostname(config)# access-list SCANSAFE_HTTPS extended deny tcp any4 object cisco1 eq 443
hostname(config)# access-list SCANSAFE_HTTPS extended deny tcp any4 object cisco2 eq 443
```

```
hostname(config)# access-list SCANSAFE_HTTP extended deny tcp any4 object dmz_network eq
443
hostname(config)# access-list SCANSAFE_HTTPS extended permit tcp any4 any4 eq 443

hostname(config)# class-map cws_class1
hostname(config-cmap)# match access-list SCANSAFE_HTTP
hostname(config)# class-map cws_class2
hostname(config-cmap)# match access-list SCANSAFE_HTTPS

hostname(config)# policy-map cws_policy
hostname(config-pmap)# class cws_class1
hostname(config-pmap-c)# inspect scansafe cws_inspect_pmap1 fail-open
hostname(config-pmap)# class cws_class2
hostname(config-pmap-c)# inspect scansafe cws_inspect_pmap2 fail-open
hostname(config)# service-policy cws_policy inside
```

Configure the User Identity Monitor

When you use identity firewall, the ASA only downloads user identity information from the AD server for users and groups included in active ACLs. The ACL must be used in a feature such as an access rule, AAA rule, service policy rule, or other feature to be considered active.

For example, although you can configure your Cloud Web Security service policy rule to use an ACL with users and groups, thus activating any relevant groups, it is not required. You could use an ACL based entirely on IP addresses.

Because Cloud Web Security can base its ScanCenter policy on user identity, you might need to download groups that are not part of an active ACL to get full identity firewall coverage for all your users. The user identity monitor lets you download group information directly from the AD agent.

Note The ASA can only monitor a maximum of 512 groups, including those configured for the user identity monitor and those monitored through active ACLs.

Procedure

Step 1 Identify the groups that you want to use in ScanCenter policies that are not already used in active ACLs. If necessary, create local user group objects.

Step 2 Download the group information from the AD agent.

user-identity monitor {**user-group** [*domain-name*****]*group-name* | **object-group-user** *object-group-name*}

```
hostname(config)# user-identity monitor user-group CISCO\\Engineering
```

Where:

- **user-group**—Specifies a group name defined in the AD server.
- **object-group-user**—The name of a local object created by the **object-group user** command. This group can include multiple groups.

Configure the Cloud Web Security Policy

After you configure the ASA service policy rules, launch the ScanCenter Portal to configure Web content scanning, filtering, malware protection services, and reports.

Go to: https://scancenter.scansafe.com/portal/admin/login.jsp.

For more information, see the Cisco ScanSafe Cloud Web Security Configuration Guides:

http://www.cisco.com/en/US/products/ps11720/products_installation_and_configuration_guides_list.html

Monitoring Cloud Web Security

To monitor Cloud Web Security, use the following commands:

- **show scansafe server**

  Shows the status of the server, whether it is the currently active server, the backup server, or unreachable.

```
hostname# show scansafe server
hostname# Primary: proxy197.scansafe.net (72.37.244.115) (REACHABLE)*
hostname# Backup: proxy137.scansafe.net (80.254.152.99)
```

- **show scansafe statistics**

  Shows information about Cloud Web Security activity, such as the number of connections redirected to the proxy server, the number of current connections being redirected, and the number of white listed connections:

```
hostname# show scansafe statistics
Current HTTP sessions : 0
Current HTTPS sessions : 0
Total HTTP Sessions : 0
Total HTTPS Sessions : 0
Total Fail HTTP sessions : 0
Total Fail HTTPS sessions : 0
Total Bytes In : 0 Bytes
Total Bytes Out : 0 Bytes
HTTP session Connect Latency in ms(min/max/avg) : 0/0/0
HTTPS session Connect Latency in ms(min/max/avg) : 0/0/0
```

- **show service policy inspect scansafe**

  Shows the number of connections that are redirected or white listed by a particular policy.

```
hostname(config)# show service-policy inspect scansafe
Global policy:
  Service-policy: global_policy
    Class-map: inspection_default
Interface inside:
  Service-policy: scansafe-pmap
    Class-map: scansafe-cmap
      Inspect: scansafe p-scansafe fail-open, packet 0, drop 0, reset-drop 0,
v6-fail-close 0
Number of whitelisted connections: 0
Number of connections allowed without scansafe inspection because of "fail-open"
config: 0
Number of connections dropped because of "fail-close" config: 0
Number of HTTP connections inspected: 0
Number of HTTPS connections inspected: 0
```

```
Number of HTTP connections dropped because of errors: 0
Number of HTTPS connections dropped because of errors: 0
```

- **show conn scansafe**

  Shows all Cloud Web Security connections, as noted by the capitol Z flag.

You can determine if a user's traffic is being redirected to the proxy servers by accessing the following URL from the client machine. The page will show a message indicating whether the user is currently using the service.

http://Whoami.scansafe.net

Examples for Cisco Cloud Web Security

Following are some examples for configuring Cloud Web Security.

- Cloud Web Security Example with Identity Firewall, page 8-15
- Active Directory Integration Example for Identity Firewall, page 8-17

Cloud Web Security Example with Identity Firewall

The following example shows a complete configuration for Cisco Cloud Web Security in single context mode, including the optional configuration for identity firewall.

Step 1 Configure Cloud Web Security on the ASA.

```
hostname(config)# scansafe general-options
hostname(cfg-scansafe)# server primary ip 192.168.115.225
hostname(cfg-scansafe)# retry-count 5
hostname(cfg-scansafe)# license 366C1D3F5CE67D33D3E9ACEC265261E5
```

Step 2 Configure identity firewall settings.

Because groups are a key feature of ScanCenter policies, you should consider enabling the identity firewall if you are not already using it. However, identity firewall is optional. The following example shows how to define the Active Directory (AD) server, the AD agent, configure identity firewall settings, and enable the user identity monitor for a few groups.

```
aaa-server AD protocol ldap
aaa-server AD (inside) host 192.168.116.220
 server-port 389
 ldap-base-dn DC=ASASCANLAB,DC=local
 ldap-scope subtree
 ldap-login-password *****
 ldap-login-dn cn=administrator,cn=Users,dc=asascanlab,dc=local
 server-type microsoft
aaa-server adagent protocol radius
 ad-agent-mode
aaa-server adagent (inside) host 192.168.116.220
 key *****
user-identity domain ASASCANLAB aaa-server AD
user-identity default-domain ASASCANLAB
user-identity action netbios-response-fail remove-user-ip
user-identity poll-import-user-group-timer hours 1
user-identity ad-agent aaa-server adagent
user-identity user-not-found enable
user-identity monitor user-group ASASCANLAB\\GROUP1
```

```
user-identity monitor user-group ASASCANLAB\\GROUPNAME
```

Step 3 (Optional) Configure a whitelist.

If there are specific users or groups you would like to exempt from Cloud Web Security filtering, you can create a whitelist.

```
class-map type inspect scansafe match-any whiteListCmap
 match user LOCAL\user1
```

Step 4 Configure ACLs.

We recommend that you split the traffic by creating separate HTTP and HTTPS class maps so that you know how many HTTP and HTTPS packets have gone through.

Then, if you need to troubleshoot you can run debug commands to distinguish how many packets have traversed each class map and find out if you are pushing through more HTTP or HTTPS traffic:

```
hostname(config)# access-list web extended permit tcp any any eq www
hostname(config)# access-list https extended permit tcp any any eq https
```

Step 5 Configure class maps.

```
hostname(config)# class-map cmap-http
hostname(config-cmap)# match access-list web

hostname(config)# class-map cmap-https
hostname(config-cmap)# match access-list https
```

Step 6 Configure inspection policy maps.

```
hostname(config)# policy-map type inspect scansafe http-pmap
hostname(config-pmap)# parameters
hostname(config-pmap-p)# default group httptraffic
hostname(config-pmap-p)# http
hostname(config-pmap-p)# class whiteListCmap
hostname(config-pmap-p)# whitelist

hostname(config)# policy-map type inspect scansafe https-pmap
hostname(config-pmap)# parameters
hostname(config-pmap-p)# default group httpstraffic
hostname(config-pmap-p)# https
hostname(config-pmap-p)# class whiteListCmap
hostname(config-pmap-p)# whitelist
```

Step 7 Configure policy maps.

The following example creates unique policy maps for Cloud Web Security traffic.

```
hostname(config)# policy-map pmap-webtraffic
hostname(config-pmap)# class cmap-http
hostname(config-pmap-c)# inspect scansafe http-pmap fail-close

hostname(config-pmap)# class cmap-https
hostname(config-pmap-c)# inspect scansafe https-pmap fail-close
```

Alternatively, you can add the classes to the default global_policy to have redirection enabled for all interfaces. Ensure that you add the classes to global_policy rather than applying a new policy map globally, or you will remove the default protocol inspections that are part of the default global policy.

```
hostname(config)# policy-map global_policy
hostname(config-pmap)# class cmap-http
hostname(config-pmap-c)# inspect scansafe http-pmap fail-close

hostname(config-pmap)# class cmap-https
hostname(config-pmap-c)# inspect scansafe https-pmap fail-close
```

Step 8 Configure service policy.

If you created a separate policy map for Cloud Web Security, the following example shows how to apply it to an interface. If you instead added the classes to the global_policy map, you are finished; you do not need to enter the service-policy command.

```
hostname(config)# service-policy pmap-webtraffic interface inside
```

Active Directory Integration Example for Identity Firewall

The following is an end-to-end example configuration for Active Directory integration. This configuration enables the identity firewall.

Procedure

Step 1 Configure the Active Directory Server Using LDAP.

The following example shows how to configure the Active Directory server on your ASA using LDAP:

```
hostname(config)# aaa-server AD protocol ldap
hostname(config-aaa-server-group)# aaa-server AD (inside) host 192.168.116.220
hostname(config-aaa-server-host)# ldap-base-dn DC=ASASCANLAB,DC=local
hostname(config-aaa-server-host)# ldap-scope subtree
hostname(config-aaa-server-host)# server-type microsoft
hostname(config-aaa-server-host)# server-port 389
hostname(config-aaa-server-host)# ldap-login-dn
cn=administrator,cn=Users,dc=asascanlab,dc=local
hostname(config-aaa-server-host)# ldap-login-password Password1
```

Step 2 Configure the Active Directory Agent Using RADIUS.

The following example shows how to configure the Active Directory Agent on your ASA using RADIUS:

```
hostname(config)# aaa-server adagent protocol radius
hostname(config-aaa-server-group)# ad-agent-mode
hostname(config-aaa-server-group)# aaa-server adagent (inside) host 192.168.116.220
hostname(config-aaa-server-host)# key cisco123
hostname(config-aaa-server-host)# user-identity ad-agent aaa-server adagent
```

Step 3 (On the AD Agent server.) Create the ASA as a Client on the AD Agent Server.

The following example shows how to create the ASA as a client on the Active Directory agent server:

```
c:\IBF\CLI\adacfg client create -name ASA5520DEVICE -ip 192.168.116.90 -secret cisco123
```

Step 4 (On the AD Agent server.) Create a Link Between the AD Agent and DCs.

The following example shows how to create a link between the Active Directory Agent and all DCs for which you want to monitor logon/logoff events:

```
c:\IBF\CLI\adacfg.exe dc create -name DCSERVER1 -host W2K3DC -domain
W2K3DC.asascanlab.local -user administrator -password Password1
c:\IBF\CLI\adacfg.exe dc list
```

Running the last command should show the status as “UP.”

For the AD_Agent to monitor logon/logoff events, you need to ensure that these are logged on all DCs that are actively being monitored. To do this, choose:

Start > Administrative Tools > Domain Controller Security Policy

Local policies > Audit Policy > Audit account logon events (success and failure)

Step 5 (Back on the ASA.) Test the AD Agent.

The following example shows how to configure the test Active Directory Agent so that it can communicate with the ASA:

```
hostname# test aaa-server ad-agent adagent
Server IP Address or name: 192.168.116.220
INFO: Attempting Ad-agent test to IP address <192.168.116.220> (timeout: 12 seconds)
INFO: Ad-agent Successful
```

See also the following command: **show user-identity ad-agent**.

Step 6 Configure the Identity Options on the ASA.

The following example shows how to configure the identity options on the ASA:

```
hostname(config)# user-identity domain ASASCANLAB aaa-server AD
hostname(config)# user-identity default-domain ASASCANLAB
```

Step 7 Configure the User Identity Options and Enabling Granular Reporting.

The following example shows how to configure the user identity options that send user credentials to the ASA and enable granular user reporting from the proxy server:

```
hostname(config)# user-identity inactive-user-timer minutes 60
hostname(config)# user-identity action netbios-response-fail remove-user-ip
hostname(config)# user-identity user-not-found enable
hostname(config)# user-identity action mac-address-mismatch remove-user-ip
hostname(config)# user-identity ad-agent active-user-database full-download
```

There are two download modes with Identify Firewall: Full download and On-demand.

- Full download—Whenever a user logs into the network, the IDFW tells the ASA the User identity immediately (recommended on the ASA 5512-X and above).
- On-demand—Whenever a user logs into the network, the ASA requests the user identity from AD.

If you are using more than one domain, then enter the following command:

```
hostname(config)# user-identity domain OTHERDOMAINNAME
```

Step 8 Monitor the Active Directory Groups.

The following example shows how to configure Active Directory groups to be monitored:

```
hostname(config)# user-identity monitor user-group ASASCANLAB\\GROUPNAME1
hostname(config)# user-identity monitor user-group ASASCANLAB\\GROUPNAME2
hostname(config)# user-identity monitor user-group ASASCANLAB\\GROUPNAME3
```

Remember to save your configuration once the above is completed.

Step 9 Download the Entire Active-User Database from the Active Directory Server.

The following command updates the specified import user group database by querying the Active Directory server immediately without waiting for the expiration of poll-import-user-group-timer:

```
hostname(config)# user-identity update import-user
```

Step 10 Download the Database from the AD Agent.

The following example shows how to manually start the download of the database from the Active Directory Agent if you think the user database is out of sync with Active Directory:

```
hostname(config)# user-identity update active-user-database
```

Step 11 Show a List of Active Users.

```
hostname# show user-identity user active list detail
```

History for Cisco Cloud Web Security

Feature Name	Platform Releases	Feature Information
Cloud Web Security	9.0(1)	This feature was introduced. Cisco Cloud Web Security provides content scanning and other malware protection service for web traffic. It can also redirect and report about web traffic based on user identity. We introduced or modified the following commands: **class-map type inspect scansafe**, **default user group**, **http**[**s**] (parameters), **inspect scansafe**, **license**, **match user group**, **policy-map type inspect scansafe**, **retry-count**, **scansafe**, **scansafe general-options**, **server** {**primary** | **backup**}, **show conn scansafe**, **show scansafe server**, **show scansafe statistics**, **user-identity monitor**, **whitelist**.

PART 2

Network Address Translation

Network Address Translation (NAT)

The following topics explain Network Address Translation (NAT) and how to configure it.

Why Use NAT?

Each computer and device within an IP network is assigned a unique IP address that identifies the host. Because of a shortage of public IPv4 addresses, most of these IP addresses are private, not routable anywhere outside of the private company network. RFC 1918 defines the private IP addresses you can use internally that should not be advertised:

- 10.0.0.0 through 10.255.255.255
- 172.16.0.0 through 172.31.255.255
- 192.168.0.0 through 192.168.255.255

One of the main functions of NAT is to enable private IP networks to connect to the Internet. NAT replaces a private IP address with a public IP address, translating the private addresses in the internal private network into legal, routable addresses that can be used on the public Internet. In this way, NAT conserves public addresses because it can be configured to advertise at a minimum only one public address for the entire network to the outside world.

Other functions of NAT include:

- Security—Keeping internal IP addresses hidden discourages direct attacks.
- IP routing solutions—Overlapping IP addresses are not a problem when you use NAT.

- Flexibility—You can change internal IP addressing schemes without affecting the public addresses available externally; for example, for a server accessible to the Internet, you can maintain a fixed IP address for Internet use, but internally, you can change the server address.
- Translating between IPv4 and IPv6 (Routed mode only) —If you want to connect an IPv6 network to an IPv4 network, NAT lets you translate between the two types of addresses.

Note NAT is not required. If you do not configure NAT for a given set of traffic, that traffic will not be translated, but will have all of the security policies applied as normal.

NAT Basics

The following topics explain some of the basics of NAT.

NAT Terminology

This document uses the following terminology:

- Real address/host/network/interface—The real address is the address that is defined on the host, before it is translated. In a typical NAT scenario where you want to translate the inside network when it accesses the outside, the inside network would be the “real” network. Note that you can translate any network connected to the ASA, not just an inside network, Therefore if you configure NAT to translate outside addresses, “real” can refer to the outside network when it accesses the inside network.
- Mapped address/host/network/interface—The mapped address is the address that the real address is translated to. In a typical NAT scenario where you want to translate the inside network when it accesses the outside, the outside network would be the “mapped” network.

  Note During address translation, IP addresses residing on the ASA’s interfaces are not translated.

- Bidirectional initiation—Static NAT allows connections to be initiated *bidirectionally*, meaning both to the host and from the host.
- Source and destination NAT—For any given packet, both the source and destination IP addresses are compared to the NAT rules, and one or both can be translated/untranslated. For static NAT, the rule is bidirectional, so be aware that “source” and “destination” are used in commands and descriptions throughout this guide even though a given connection might originate at the “destination” address.

NAT Types

You can implement NAT using the following methods:

- Dynamic NAT—A group of real IP addresses are mapped to a (usually smaller) group of mapped IP addresses, on a first come, first served basis. Only the real host can initiate traffic. See Dynamic NAT, page 9-12.
- Dynamic Port Address Translation (PAT)—A group of real IP addresses are mapped to a single IP address using a unique source port of that IP address. See Dynamic PAT, page 9-18.
- Static NAT—A consistent mapping between a real and mapped IP address. Allows bidirectional traffic initiation. See Static NAT, page 9-27.
- Identity NAT—A real address is statically translated to itself, essentially bypassing NAT. You might want to configure NAT this way when you want to translate a large group of addresses, but then want to exempt a smaller subset of addresses. See Identity NAT, page 9-37.

Network Object NAT and Twice NAT

The ASA can implement address translation in two ways: *network object NAT* and *twice NAT*.

We recommend using network object NAT unless you need the extra features that twice NAT provides. Network object NAT is easier to configure, and might be more reliable for applications such as Voice over IP (VoIP). (For VoIP, because twice NAT is applicable only between two objects, you might see a failure in the translation of indirect addresses that do not belong to either of the objects.)

Network Object NAT

All NAT rules that are configured as a parameter of a network object are considered to be *network object NAT* rules. Network object NAT is a quick and easy way to configure NAT for a network object, which can be a single IP address, a range of addresses, or a subnet.

After you configure the network object, you can then identify the mapped address for that object, either as an inline address or as another network object or network object group.

When a packet enters the ASA, both the source and destination IP addresses are checked against the network object NAT rules. The source and destination address in the packet can be translated by separate rules if separate matches are made. These rules are not tied to each other; different combinations of rules can be used depending on the traffic.

Because the rules are never paired, you cannot specify that sourceA/destinationA should have a different translation than sourceA/destinationB. Use twice NAT for that kind of functionality (twice NAT lets you identify the source and destination address in a single rule).

Twice NAT

Twice NAT lets you identify both the source and destination address in a single rule. Specifying both the source and destination addresses lets you specify that sourceA/destinationA can have a different translation than sourceA/destinationB.

Note For static NAT, the rule is bidirectional, so be aware that "source" and "destination" are used in commands and descriptions throughout this guide even though a given connection might originate at the "destination" address. For example, if you configure static NAT with port address translation, and specify the source address as a Telnet server, and you want all traffic going to that Telnet server to have the port translated from 2323 to 23, then in the command, you must specify the *source* ports to be translated (real: 23, mapped: 2323). You specify the source ports because you specified the Telnet server address as the source address.

The destination address is optional. If you specify the destination address, you can either map it to itself (identity NAT), or you can map it to a different address. The destination mapping is always a static mapping.

Twice NAT also lets you use service objects for static NAT with port translation; network object NAT only accepts inline definition.

Comparing Network Object NAT and Twice NAT

The main differences between these two NAT types are:

- How you define the real address.
  - Network object NAT—You define NAT as a parameter for a network object. A network object names an IP host, range, or subnet so you can then use the object in the NAT configuration instead of the actual IP addresses. The network object IP address serves as the real address. This method lets you easily add NAT to network objects that might already be used in other parts of your configuration.
  - Twice NAT—You identify a network object or network object group for both the real and mapped addresses. In this case, NAT is not a parameter of the network object; the network object or group is a parameter of the NAT configuration. The ability to use a network object *group* for the real address means that twice NAT is more scalable.
- How source and destination NAT is implemented.
  - Network object NAT— Each rule can apply to either the source or destination of a packet. So two rules might be used, one for the source IP address, and one for the destination IP address. These two rules cannot be tied together to enforce a specific translation for a source/destination combination.
  - Twice NAT—A single rule translates both the source and destination. A matching packet only matches the one rule, and further rules are not checked. Even if you do not configure the optional destination address for twice NAT, a matching packet still only matches one twice NAT rule. The source and destination are tied together, so you can enforce different translations depending on the source/destination combination. For example, sourceA/destinationA can have a different translation than sourceA/destinationB.
- Order of NAT Rules.
  - Network object NAT—Automatically ordered in the NAT table.
  - Twice NAT—Manually ordered in the NAT table (before or after network object NAT rules).

NAT Rule Order

Network object NAT rules and twice NAT rules are stored in a single table that is divided into three sections. Section 1 rules are applied first, then section 2, and finally section 3, until a match is found. For example, if a match is found in section 1, sections 2 and 3 are not evaluated. The following table shows the order of rules within each section.

Table 9-1 NAT Rule Table

Table Section	Rule Type	Order of Rules within the Section
Section 1	Twice NAT	Applied on a first match basis, in the order they appear in the configuration. Because the first match is applied, you must ensure that specific rules come before more general rules, or the specific rules might not be applied as desired. By default, twice NAT rules are added to section 1. **Note** If you configure EasyVPN remote, the ASA dynamically adds invisible NAT rules to the end of this section. Be sure that you do not configure a twice NAT rule in this section that might match your VPN traffic, instead of matching the invisible rule. If VPN does not work due to NAT failure, consider adding twice NAT rules to section 3 instead.
Section 2	Network object NAT	If a match in section 1 is not found, section 2 rules are applied in the following order, as automatically determined by the ASA: 1. Static rules. 2. Dynamic rules. **Within each rule type, the following ordering guidelines are used:** 1. Quantity of real IP addresses—From smallest to largest. For example, an object with one address will be assessed before an object with 10 addresses. 2. For quantities that are the same, then the IP address number is used, from lowest to highest. For example, 10.1.1.0 is assessed before 11.1.1.0. 3. If the same IP address is used, then the name of the network object is used, in alphabetical order. For example, abracadabra is assessed before catwoman.
Section 3	Twice NAT	If a match is still not found, section 3 rules are applied on a first match basis, in the order they appear in the configuration. This section should contain your most general rules. You must also ensure that any specific rules in this section come before general rules that would otherwise apply. You can specify whether to add a twice NAT rule to section 3 when you add the rule.

For section 2 rules, for example, you have the following IP addresses defined within network objects:

192.168.1.0/24 (static)

192.168.1.0/24 (dynamic)

10.1.1.0/24 (static)

192.168.1.1/32 (static)

172.16.1.0/24 (dynamic) (object def)

172.16.1.0/24 (dynamic) (object abc)

The resultant ordering would be:

192.168.1.1/32 (static)

10.1.1.0/24 (static)

192.168.1.0/24 (static)

172.16.1.0/24 (dynamic) (object abc)

172.16.1.0/24 (dynamic) (object def)

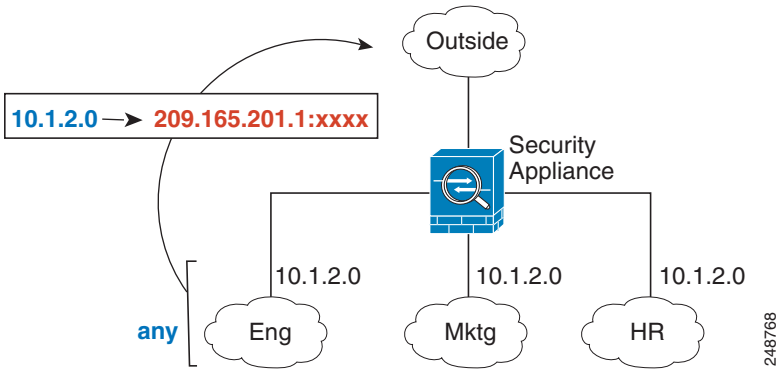
192.168.1.0/24 (dynamic)

NAT Interfaces

In routed mode, you can configure a NAT rule to apply to any interface (in other words, all interfaces), or you can identify specific real and mapped interfaces. You can also specify any interface for the real address, and a specific interface for the mapped address, or vice versa.

For example, you might want to specify any interface for the real address and specify the outside interface for the mapped address if you use the same private addresses on multiple interfaces, and you want to translate them all to the same global pool when accessing the outside.

Figure 9-1 Specifying Any Interface



In transparent mode, you must choose specific source and destination interfaces.

Guidelines for NAT

The following topics provide detailed guidelines for implementing NAT.

- Firewall Mode Guidelines for NAT, page 9-7
- IPv6 NAT Guidelines, page 9-7

Firewall Mode Guidelines for NAT

NAT is supported in routed and transparent firewall mode. However, transparent mode has the following restrictions:

- In transparent mode, you must specify the real and mapped interfaces; you cannot specify “any” as the interface.
- In transparent mode, you cannot configure interface PAT, because the transparent mode interfaces do not have IP addresses. You also cannot use the management IP address as a mapped address.
- In transparent mode, translating between IPv4 and IPv6 networks is not supported. Translating between two IPv6 networks, or between two IPv4 networks is supported.

IPv6 NAT Guidelines

NAT supports IPv6 with the following guidelines and restrictions.

- For routed mode, you can also translate between IPv4 and IPv6.
- For transparent mode, translating between IPv4 and IPv6 networks is not supported. Translating between two IPv6 networks, or between two IPv4 networks is supported.
- For transparent mode, a PAT pool is not supported for IPv6.
- For static NAT, you can specify an IPv6 subnet up to /64. Larger subnets are not supported.
- When using FTP with NAT46, when an IPv4 FTP client connects to an IPv6 FTP server, the client must use either the extended passive mode (EPSV) or extended port mode (EPRT); PASV and PORT commands are not supported with IPv6.

IPv6 NAT Recommendations

You can use NAT to translate between IPv6 networks, and also to translate between IPv4 and IPv6 networks (routed mode only). We recommend the following best practices:

- NAT66 (IPv6-to-IPv6)—We recommend using static NAT. Although you can use dynamic NAT or PAT, IPv6 addresses are in such large supply, you do not have to use dynamic NAT. If you do not want to allow returning traffic, you can make the static NAT rule unidirectional (twice NAT only).
- NAT46 (IPv4-to-IPv6)—We recommend using static NAT. Because the IPv6 address space is so much larger than the IPv4 address space, you can easily accommodate a static translation. If you do not want to allow returning traffic, you can make the static NAT rule unidirectional (twice NAT only). When translating to an IPv6 subnet (/96 or lower), the resulting mapped address is by default an IPv4-embedded IPv6 address, where the 32-bits of the IPv4 address is embedded after the IPv6 prefix. For example, if the IPv6 prefix is a /96 prefix, then the IPv4 address is appended in the last 32-bits of the address. For example, if you map 192.168.1.0/24 to 201b::0/96, then 192.168.1.4 will

be mapped to 201b::0.192.168.1.4 (shown with mixed notation). If the prefix is smaller, such as /64, then the IPv4 address is appended after the prefix, and a suffix of 0s is appended after the IPv4 address. You can also optionally translate the addresses net-to-net, where the first IPv4 address maps to the first IPv6 address, the second to the second, and so on.

- NAT64 (IPv6-to-IPv4)—You may not have enough IPv4 addresses to accommodate the number of IPv6 addresses. We recommend using a dynamic PAT pool to provide a large number of IPv4 translations.

Additional Guidelines for NAT

- (Network object NAT only.) You can only define a single NAT rule for a given object; if you want to configure multiple NAT rules for an object, you need to create multiple objects with different names that specify the same IP address, for example, **object network obj-10.10.10.1-01**, **object network obj-10.10.10.1-02**, and so on.
- (Twice NAT only.) You cannot configure FTP destination port translation when the source IP address is a subnet (or any other application that uses a secondary connection); the FTP data channel establishment does not succeed. For example, the following configuration does not work:

```
object network MyInsNet
 subnet 10.1.2.0 255.255.255.0
object network MapInsNet
 subnet 209.165.202.128 255.255.255.224
object network Server1
 host 209.165.200.225
object network Server1_mapped
 host 10.1.2.67
object service REAL_ftp
 service tcp destination eq ftp
object service MAPPED_ftp
 service tcp destination eq 2021
object network MyOutNet
 subnet 209.165.201.0 255.255.255.224

nat (inside,outside) source static MyInsNet MapInsNet destination static
Server1_mapped Server1 service MAPPED_ftp REAL_ftp
```

- If you change the NAT configuration, and you do not want to wait for existing translations to time out before the new NAT configuration is used, you can clear the translation table using the **clear xlate** command. However, clearing the translation table disconnects all current connections that use translations.

Note If you remove a dynamic NAT or PAT rule, and then add a new rule with mapped addresses that overlap the addresses in the removed rule, then the new rule will not be used until all connections associated with the removed rule time out or are cleared using the **clear xlate** command. This safeguard ensures that the same address is not assigned to multiple hosts.

- Objects and object groups used in NAT cannot be undefined; they must include IP addresses.
- You cannot use an object group with both IPv4 and IPv6 addresses; the object group must include only one type of address.
- (Twice NAT only.) When using the **any** keyword in a NAT rule, the definition of “any” traffic (IPv4 vs. IPv6) depends on the rule. Before the ASA performs NAT on a packet, the packet must be IPv6-to-IPv6 or IPv4-to-IPv4; with this prerequisite, the ASA can determine the value of **any** in a NAT rule. For example, if you configure a rule from “any” to an IPv6 server, and that server was

mapped from an IPv4 address, then **any** means “any IPv6 traffic.” If you configure a rule from “any” to “any,” and you map the source to the interface IPv4 address, then **any** means “any IPv4 traffic” because the mapped interface address implies that the destination is also IPv4.

- You can use the same mapped object or group in multiple NAT rules.
- The mapped IP address pool cannot include:
  - The mapped interface IP address. If you specify “any” interface for the rule, then all interface IP addresses are disallowed. For interface PAT (routed mode only), use the **interface** keyword instead of the IP address.
  - (Transparent mode) The management IP address.
  - (Dynamic NAT) The standby interface IP address when VPN is enabled.
  - Existing VPN pool addresses.
- Avoid using overlapping addresses in static and dynamic NAT policies. For example, with overlapping addresses, a PPTP connection can fail to get established if the secondary connection for PPTP hits the static instead of dynamic xlate.
- For application inspection limitations with NAT or PAT, see Default Inspections and NAT Limitations, page 12-6.
- The default behavior for identity NAT has proxy ARP enabled, matching other static NAT rules. You can disable proxy ARP if desired. See Routing NAT Packets, page 10-11 for more information.
- If you specify an optional interface, then the ASA uses the NAT configuration to determine the egress interface, but you have the option to always use a route lookup instead. See Routing NAT Packets, page 10-11 for more information.
- You can improve system performance and reliability by using the transactional commit model for NAT. See the basic settings chapter in the general operations configuration guide for more information. Use the **asp rule-engine transactional-commit nat** command.

Network Object NAT Guidelines for Mapped Address Objects

For dynamic NAT, you must use an object or group for the mapped addresses. For the other NAT types, you can use an object or group, or you have the option of using inline addresses. Network object groups are particularly useful for creating a mapped address pool with discontinuous IP address ranges or multiple hosts or subnets. Use the **object network** and **object-group network** commands to create the objects.

Consider the following guidelines when creating objects for mapped addresses.

- A network object group can contain objects or inline addresses of either IPv4 or IPv6 addresses. The group cannot contain both IPv4 and IPv6 addresses; it must contain one type only.
- See Additional Guidelines for NAT, page 9-8 for information about disallowed mapped IP addresses.
- Dynamic NAT:
  - You cannot use an inline address; you must configure a network object or group.
  - The object or group cannot contain a subnet; the object must define a range; the group can include hosts and ranges.
  - If a mapped network object contains both ranges and host IP addresses, then the ranges are used for dynamic NAT, and then the host IP addresses are used as a PAT fallback.
- Dynamic PAT (Hide):

  - Instead of using an object, you can optionally configure an inline host address or specify the interface address.
  - If you use an object, the object or group cannot contain a subnet. The object must define a host, or for a PAT pool, a range. The group (for a PAT pool) can include hosts and ranges.
- Static NAT or Static NAT with port translation:
  - Instead of using an object, you can configure an inline address or specify the interface address (for static NAT-with-port-translation).
  - If you use an object, the object or group can contain a host, range, or subnet.
- Identity NAT
  - Instead of using an object, you can configure an inline address.
  - If you use an object, the object must match the real addresses you want to translate.

Twice NAT Guidelines for Real and Mapped Address Objects

For each NAT rule, configure up to four network objects or groups for:

- **Source real address**
- **Source mapped address**
- **Destination real address**
- **Destination mapped address**

Objects are required unless you specify the **any** keyword inline to represent all traffic, or for some types of NAT, the **interface** keyword to represent the interface address. Network object groups are particularly useful for creating a mapped address pool with discontinuous IP address ranges or multiple hosts or subnets. Use the **object network** and **object-group network** commands to create the objects.

Consider the following guidelines when creating objects for twice NAT.

- A network object group can contain objects or inline addresses of either IPv4 or IPv6 addresses. The group cannot contain both IPv4 and IPv6 addresses; it must contain one type only.
- See Additional Guidelines for NAT, page 9-8 for information about disallowed mapped IP addresses.
- Source Dynamic NAT:
  - You typically configure a larger group of real addresses to be mapped to a smaller group.
  - The mapped object or group cannot contain a subnet; the object must define a range; the group can include hosts and ranges.
  - If a mapped network object contains both ranges and host IP addresses, then the ranges are used for dynamic NAT, and the host IP addresses are used as a PAT fallback.
- Source Dynamic PAT (Hide):
  - If you use an object, the object or group cannot contain a subnet. The object must define a host, or for a PAT pool, a range. The group (for a PAT pool) can include hosts and ranges.
- Source Static NAT or Static NAT with port translation:
  - The mapped object or group can contain a host, range, or subnet.
  - The static mapping is typically one-to-one, so the real addresses have the same quantity as the mapped addresses. You can, however, have different quantities if desired.

- Source Identity NAT
  - The real and mapped objects must match. You can use the same object for both, or you can create separate objects that contain the same IP addresses.
- Destination Static NAT or Static NAT with port translation (the destination translation is always static):
  - Although the main feature of twice NAT is the inclusion of the destination IP address, the destination address is optional. If you do specify the destination address, you can configure static translation for that address or just use identity NAT for it. You might want to configure twice NAT without a destination address to take advantage of some of the other qualities of twice NAT, including the use of network object groups for real addresses, or manually ordering of rules. For more information, see Comparing Network Object NAT and Twice NAT, page 9-4.
  - For identity NAT, the real and mapped objects must match. You can use the same object for both, or you can create separate objects that contain the same IP addresses.
  - The static mapping is typically one-to-one, so the real addresses have the same quantity as the mapped addresses. You can, however, have different quantities if desired.
  - For static interface NAT with port translation (routed mode only), you can specify the **interface** keyword instead of a network object/group for the mapped address.

Twice NAT Guidelines for Service Objects for Real and Mapped Ports

You can optionally configure service objects for:

- **Source real port (Static only)** *or* **Destination real port**
- **Source mapped port (Static only)** *or* **Destination mapped port**

Use the **object service** command to create the objects.

Consider the following guidelines when creating objects for twice NAT.

- NAT only supports TCP or UDP. When translating a port, be sure the protocols in the real and mapped service objects are identical (both TCP or both UDP).
- The “not equal” (**neq**) operator is not supported.
- For identity port translation, you can use the same service object for both the real and mapped ports.
- Source Dynamic NAT—Source Dynamic NAT does not support port translation.
- Source Dynamic PAT (Hide)—Source Dynamic PAT does not support port translation.
- Source Static NAT, Static NAT with port translation, or Identity NAT—A service object can contain both a source and destination port; however, you should specify *either* the source *or* the destination port for both service objects. You should only specify *both* the source and destination ports if your application uses a fixed source port (such as some DNS servers); but fixed source ports are rare. For example, if you want to translate the port for the source host, then configure the source service.
- Destination Static NAT or Static NAT with port translation (the destination translation is always static)—For non-static source NAT, you can only perform port translation on the destination. A service object can contain both a source and destination port, but only the destination port is used in this case. If you specify the source port, it will be ignored.

Dynamic NAT

The following topics explain dynamic NAT and how to configure it.

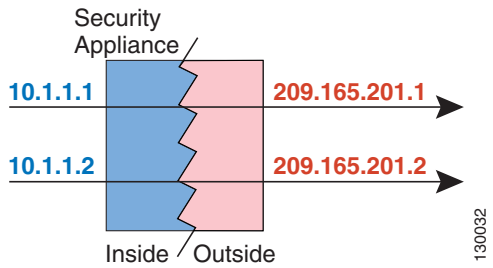
About Dynamic NAT

Dynamic NAT translates a group of real addresses to a pool of mapped addresses that are routable on the destination network. The mapped pool typically includes fewer addresses than the real group. When a host you want to translate accesses the destination network, the ASA assigns the host an IP address from the mapped pool. The translation is created only when the real host initiates the connection. The translation is in place only for the duration of the connection, and a given user does not keep the same IP address after the translation times out. Users on the destination network, therefore, cannot initiate a reliable connection to a host that uses dynamic NAT, even if the connection is allowed by an access rule.

Note For the duration of the translation, a remote host can initiate a connection to the translated host if an access rule allows it. Because the address is unpredictable, a connection to the host is unlikely. Nevertheless, in this case you can rely on the security of the access rule.

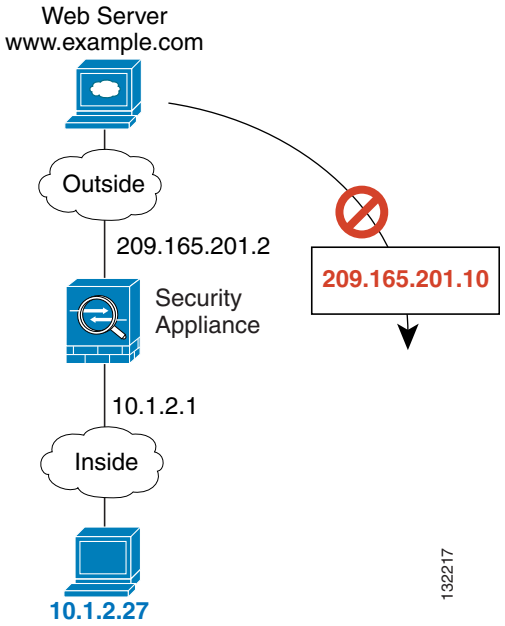
The following figure shows a typical dynamic NAT scenario. Only real hosts can create a NAT session, and responding traffic is allowed back.

Figure 9-2 Dynamic NAT



The following figure shows a remote host attempting to initiate a connection to a mapped address. This address is not currently in the translation table; therefore, the ASA drops the packet.

Figure 9-3 Remote Host Attempts to Initiate a Connection to a Mapped Address



Dynamic NAT Disadvantages and Advantages

Dynamic NAT has these disadvantages:

- If the mapped pool has fewer addresses than the real group, you could run out of addresses if the amount of traffic is more than expected.

  Use PAT or a PAT fall-back method if this event occurs often because PAT provides over 64,000 translations using ports of a single address.

- You have to use a large number of routable addresses in the mapped pool, and routable addresses may not be available in large quantities.

The advantage of dynamic NAT is that some protocols cannot use PAT. PAT does not work with the following:

- IP protocols that do not have a port to overload, such as GRE version 0.
- Some multimedia applications that have a data stream on one port, the control path on another port, and are not open standard.

See Default Inspections and NAT Limitations, page 12-6 for more information about NAT and PAT support.

Configure Dynamic Network Object NAT

This section describes how to configure network object NAT for dynamic NAT.

Procedure

Step 1 Create a host or range network object (**object network** command), or a network object group (**object-group network** command), for the mapped addresses.

- The object or group cannot contain a subnet; the object must define a range; the group can include hosts and ranges.
- If a mapped network object contains both ranges and host IP addresses, then the ranges are used for dynamic NAT, and then the host IP addresses are used as a PAT fallback.

Step 2 Create or edit the network object for which you want to configure NAT.

object network *obj_name*

Example

```
hostname(config)# object network my-host-obj1
```

Step 3 (Skip when editing an object that has the right address.) Define the real IPv4 or IPv6 addresses that you want to translate.

- **host** {*IPv4_address* | *IPv6_address*}—The IPv4 or IPv6 address of a single host. For example, 10.1.1.1 or 2001:DB8::0DB8:800:200C:417A.
- **subnet** {*IPv4_address IPv4_mask* | *IPv6_address*/*IPv6_prefix*}—The address of a network. For IPv4 subnets, include the mask after a space, for example, 10.0.0.0 255.0.0.0. For IPv6, include the address and prefix as a single unit (no spaces), such as 2001:DB8:0:CD30::/60.
- **range** *start_address end_address*—A range of addresses. You can specify IPv4 or IPv6 ranges. Do not include masks or prefixes.

Example

```
hostname(config-network-object)# host 10.2.2.2
```

Step 4 Configure **dynamic NAT** for the object IP addresses. You can only define a single NAT rule for a given object.

nat [**(***real_ifc*,*mapped_ifc***)**] **dynamic** *mapped_obj* [**interface** [**ipv6**]] [**dns**]

Example

```
hostname(config-network-object)# nat (inside,outside) dynamic MAPPED_IPS interface
```

Where:

- Interfaces—(Required for transparent mode) Specify the real (*real_ifc*) and mapped (*mapped_ifc*) interfaces. Be sure to include the parentheses. In routed mode, if you do not specify the real and mapped interfaces, all interfaces are used. You can also specify the keyword **any** for one or both of the interfaces, for example (any,outside).
- Mapped IP address—Specify the network object or network object group that includes the mapped IP addresses.

- Interface PAT fallback—(Optional) The **interface** keyword enables interface PAT fallback. After the mapped IP addresses are used up, then the IP address of the mapped interface is used. If you specify **ipv6**, then the IPv6 address of the interface is used. For this option, you must configure a specific interface for the *mapped_ifc*. (You cannot specify **interface** in transparent mode).
- DNS—(Optional) The **dns** keyword translates DNS replies. Be sure DNS inspection is enabled (it is enabled by default). See DNS and NAT, page 10-21 for more information.

Examples

The following example configures dynamic NAT that hides the 192.168.2.0 network behind a range of outside addresses 10.2.2.1 through 10.2.2.10:

```
hostname(config)# object network my-range-obj
hostname(config-network-object)# range 10.2.2.1 10.2.2.10
hostname(config)# object network my-inside-net
hostname(config-network-object)# subnet 192.168.2.0 255.255.255.0
hostname(config-network-object)# nat (inside,outside) dynamic my-range-obj
```

The following example configures dynamic NAT with dynamic PAT backup. Hosts on inside network 10.76.11.0 are mapped first to the nat-range1 pool (10.10.10.10-10.10.10.20). After all addresses in the nat-range1 pool are allocated, dynamic PAT is performed using the pat-ip1 address (10.10.10.21). In the unlikely event that the PAT translations are also used up, dynamic PAT is performed using the outside interface address.

```
hostname(config)# object network nat-range1
hostname(config-network-object)# range 10.10.10.10 10.10.10.20

hostname(config-network-object)# object network pat-ip1
hostname(config-network-object)# host 10.10.10.21

hostname(config-network-object)# object-group network nat-pat-grp
hostname(config-network-object)# network-object object nat-range1
hostname(config-network-object)# network-object object pat-ip1

hostname(config-network-object)# object network my_net_obj5
hostname(config-network-object)# subnet 10.76.11.0 255.255.255.0
hostname(config-network-object)# nat (inside,outside) dynamic nat-pat-grp interface
```

The following example configures dynamic NAT with dynamic PAT backup to translate IPv6 hosts to IPv4. Hosts on inside network 2001:DB8::/96 are mapped first to the IPv4_NAT_RANGE pool (209.165.201.1 to 209.165.201.30). After all addresses in the IPv4_NAT_RANGE pool are allocated, dynamic PAT is performed using the IPv4_PAT address (209.165.201.31). In the event that the PAT translations are also used up, dynamic PAT is performed using the outside interface address.

```
hostname(config)# object network IPv4_NAT_RANGE
hostname(config-network-object)# range 209.165.201.1 209.165.201.30

hostname(config-network-object)# object network IPv4_PAT
hostname(config-network-object)# host 209.165.201.31

hostname(config-network-object)# object-group network IPv4_GROUP
hostname(config-network-object)# network-object object IPv4_NAT_RANGE
hostname(config-network-object)# network-object object IPv4_PAT

hostname(config-network-object)# object network my_net_obj5
hostname(config-network-object)# subnet 2001:DB8::/96
hostname(config-network-object)# nat (inside,outside) dynamic IPv4_GROUP interface
```

Configure Dynamic Twice NAT

This section describes how to configure twice NAT for dynamic NAT.

Procedure

Step 1 Create host or range network objects (**object network** command), or network object groups (**object-group network** command), for the source real addresses, the source mapped addresses, the destination real addresses, and the destination mapped addresses.

- If you want to translate all source traffic, you can skip adding an object for the source real addresses, and instead specify the **any** keyword in the **nat** command.
- If you want to configure destination static interface NAT with port translation only, you can skip adding an object for the destination mapped addresses, and instead specify the **interface** keyword in the **nat** command.

If you do create objects, consider the following guidelines:

- You typically configure a larger group of real addresses to be mapped to a smaller group.
- The object or group cannot contain a subnet; the object must define a range; the group can include hosts and ranges.
- If a mapped network object contains both ranges and host IP addresses, then the ranges are used for dynamic NAT, and then the host IP addresses are used as a PAT fallback.

Step 2 (Optional.) Create service objects for the destination real ports and the destination mapped ports.

For dynamic NAT, you can only perform port translation on the destination. A service object can contain both a source and destination port, but only the destination port is used in this case. If you specify the source port, it will be ignored.

Step 3 Configure **dynamic NAT**.

```
nat [(real_ifc,mapped_ifc)] [line | {after-auto [line]}]
source dynamic {real_obj | any}
{mapped_obj [interface [ipv6]]}
[destination static {mapped_obj | interface [ipv6]} real_obj]
[service mapped_dest_svc_obj real_dest_svc_obj]
[dns] [unidirectional] [inactive] [description desc]
```

Example

```
hostname(config)# nat (inside,outside) source dynamic MyInsNet NAT_POOL
destination static Server1_mapped Server1 service MAPPED_SVC REAL_SVC
```

Where:

- Interfaces—(Required for transparent mode) Specify the real (*real_ifc*) and mapped (*mapped_ifc*) interfaces. Be sure to include the parentheses. In routed mode, if you do not specify the real and mapped interfaces, all interfaces are used. You can also specify the keyword **any** for one or both of the interfaces, for example (any,outside).
- Section and Line—(Optional.) By default, the NAT rule is added to the end of section 1 of the NAT table (see NAT Rule Order, page 9-5). If you want to add the rule into section 3 instead (after the network object NAT rules), then use the **after-auto** keyword. You can insert a rule anywhere in the applicable section using the *line* argument.
- Source addresses:
  - Real—Specify a network object, group, or the **any** keyword.

  - Mapped—Specify a different network object or group. You can optionally configure the following fallback method:

    Interface PAT fallback—(Routed mode only) The **interface** keyword enables interface PAT fallback. If you specify **ipv6**, then the IPv6 address of the interface is used. After the mapped IP addresses are used up, then the IP address of the mapped interface is used. For this option, you must configure a specific interface for the *mapped_ifc*.

- Destination addresses (Optional):
  - Mapped—Specify a network object or group, or for static interface NAT with port translation only, specify the **interface** keyword. If you specify **ipv6**, then the IPv6 address of the interface is used. If you specify **interface**, be sure to also configure the **service** keyword. For this option, you must configure a specific interface for the *real_ifc*. See Static Interface NAT with Port Translation, page 9-29 for more information.
  - Real—Specify a network object or group. For identity NAT, simply use the same object or group for both the real and mapped addresses.
- Destination port—(Optional.) Specify the **service** keyword along with the mapped and real service objects. For identity port translation, simply use the same service object for both the real and mapped ports.
- DNS—(Optional; for a source-only rule.) The **dns** keyword translates DNS replies. Be sure DNS inspection is enabled (it is enabled by default). You cannot configure the **dns** keyword if you configure a **destination** address. See DNS and NAT, page 10-21 for more information.
- Unidirectional—(Optional.) Specify **unidirectional** so the destination addresses cannot initiate traffic to the source addresses.
- Inactive—(Optional.) To make this rule inactive without having to remove the command, use the **inactive** keyword. To reactivate it, reenter the whole command without the **inactive** keyword.
- Description—Optional.) Provide a description up to 200 characters using the **description** keyword.

Examples

The following example configures dynamic NAT for inside network 10.1.1.0/24 when accessing servers on the 209.165.201.1/27 network as well as servers on the 203.0.113.0/24 network:

```
hostname(config)# object network INSIDE_NW
hostname(config-network-object)# subnet 10.1.1.0 255.255.255.0

hostname(config)# object network MAPPED_1
hostname(config-network-object)# range 209.165.200.225 209.165.200.254

hostname(config)# object network MAPPED_2
hostname(config-network-object)# range 209.165.202.129 209.165.200.158

hostname(config)# object network SERVERS_1
hostname(config-network-object)# subnet 209.165.201.0 255.255.255.224

hostname(config)# object network SERVERS_2
hostname(config-network-object)# subnet 203.0.113.0 255.255.255.0

hostname(config)# nat (inside,outside) source dynamic INSIDE_NW MAPPED_1 destination
static SERVERS_1 SERVERS_1
hostname(config)# nat (inside,outside) source dynamic INSIDE_NW MAPPED_2 destination
static SERVERS_2 SERVERS_2
```

The following example configures dynamic NAT for an IPv6 inside network 2001:DB8:AAAA::/96 when accessing servers on the IPv4 209.165.201.1/27 network as well as servers on the 203.0.113.0/24 network:

```
hostname(config)# object network INSIDE_NW
hostname(config-network-object)# subnet 2001:DB8:AAAA::/96

hostname(config)# object network MAPPED_1
hostname(config-network-object)# range 209.165.200.225 209.165.200.254

hostname(config)# object network MAPPED_2
hostname(config-network-object)# range 209.165.202.129 209.165.200.158

hostname(config)# object network SERVERS_1
hostname(config-network-object)# subnet 209.165.201.0 255.255.255.224

hostname(config)# object network SERVERS_2
hostname(config-network-object)# subnet 203.0.113.0 255.255.255.0

hostname(config)# nat (inside,outside) source dynamic INSIDE_NW MAPPED_1 destination
static SERVERS_1 SERVERS_1
hostname(config)# nat (inside,outside) source dynamic INSIDE_NW MAPPED_2 destination
static SERVERS_2 SERVERS_2
```

Dynamic PAT

The following topics describe dynamic PAT.

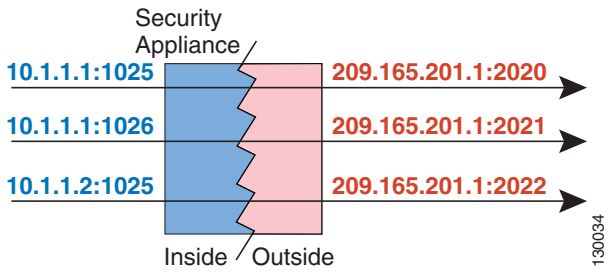
About Dynamic PAT

Dynamic PAT translates multiple real addresses to a single mapped IP address by translating the real address and source port to the mapped address and a unique port. If available, the real source port number is used for the mapped port. However, if the real port is *not* available, by default the mapped ports are chosen from the same range of ports as the real port number: 0 to 511, 512 to 1023, and 1024 to 65535. Therefore, ports below 1024 have only a small PAT pool that can be used. If you have a lot of traffic that uses the lower port ranges, you can specify a flat range of ports to be used instead of the three unequal-sized tiers.

Each connection requires a separate translation session because the source port differs for each connection. For example, 10.1.1.1:1025 requires a separate translation from 10.1.1.1:1026.

The following figure shows a typical dynamic PAT scenario. Only real hosts can create a NAT session, and responding traffic is allowed back. The mapped address is the same for each translation, but the port is dynamically assigned.

Figure 9-4 Dynamic PAT



After the connection expires, the port translation also expires. For multi-session PAT, the PAT timeout is used, 30 seconds by default. For per-session PAT, the xlate is immediately removed. Users on the destination network cannot reliably initiate a connection to a host that uses PAT (even if the connection is allowed by an access rule).

Note For the duration of the translation, a remote host can initiate a connection to the translated host if an access rule allows it. Because the port address (both real and mapped) is unpredictable, a connection to the host is unlikely. Nevertheless, in this case you can rely on the security of the access rule.

Dynamic PAT Disadvantages and Advantages

Dynamic PAT lets you use a single mapped address, thus conserving routable addresses. You can even use the ASA interface IP address as the PAT address.

Dynamic PAT does not work with some multimedia applications that have a data stream that is different from the control path. See Default Inspections and NAT Limitations, page 12-6 for more information about NAT and PAT support.

Dynamic PAT might also create a large number of connections appearing to come from a single IP address, and servers might interpret the traffic as a DoS attack. You can configure a PAT pool of addresses and use a round-robin assignment of PAT addresses to mitigate this situation.

PAT Pool Object Guidelines

When creating network objects for a PAT pool, follow these guidelines.

For a PAT pool

- If available, the real source port number is used for the mapped port. However, if the real port is *not* available, by default the mapped ports are chosen from the same range of ports as the real port number: 0 to 511, 512 to 1023, and 1024 to 65535. Therefore, ports below 1024 have only a small PAT pool that can be used. (8.4(3) and later, not including 8.5(1) or 8.6(1)) If you have a lot of traffic that uses the lower port ranges, you can specify a flat range of ports to be used instead of the three unequal-sized tiers: either 1024 to 65535, or 1 to 65535.
- If you use the same PAT pool object in two separate rules, then be sure to specify the same options for each rule. For example, if one rule specifies extended PAT and a flat range, then the other rule must also specify extended PAT and a flat range.

For extended PAT for a PAT pool

- Many application inspections do not support extended PAT. See Default Inspections and NAT Limitations, page 12-6 for a complete list of unsupported inspections.
- If you enable extended PAT for a dynamic PAT rule, then you cannot also use an address in the PAT pool as the PAT address in a separate static NAT with port translation rule. For example, if the PAT pool includes 10.1.1.1, then you cannot create a static NAT-with-port-translation rule using 10.1.1.1 as the PAT address.
- If you use a PAT pool and specify an interface for fallback, you cannot specify extended PAT.
- For VoIP deployments that use ICE or TURN, do not use extended PAT. ICE and TURN rely on the PAT binding to be the same for all destinations.

For round robin for a PAT pool

- If a host has an existing connection, then subsequent connections from that host will use the same PAT IP address if ports are available. **Note**: This "stickiness" does not survive a failover. If the ASA fails over, then subsequent connections from a host may not use the initial IP address.
- Round robin, especially when combined with extended PAT, can consume a large amount of memory. Because NAT pools are created for every mapped protocol/IP address/port range, round robin results in a large number of concurrent NAT pools, which use memory. Extended PAT results in an even larger number of concurrent NAT pools.

Configure Dynamic Network Object PAT

This section describes how to configure network object NAT for dynamic PAT.

Procedure

Step 1 (Optional.) Create a host or range network object (**object network** command), or a network object group (**object-group network** command), for the mapped addresses.

- Instead of using an object, you can optionally configure an inline host address or specify the interface address.
- If you use an object, the object or group cannot contain a subnet; the object must define a host, or for a PAT pool, a range; the group (for a PAT pool) can include hosts and ranges.

Step 2 Create or edit the network object for which you want to configure NAT.

```
object network obj_name
```

Example

```
hostname(config)# object network my-host-obj1
```

Step 3 (Skip when editing an object that has the right address.) Define the real IPv4 or IPv6 addresses that you want to translate.

- **host** {*IPv4_address* | *IPv6_address*}—The IPv4 or IPv6 address of a single host. For example, 10.1.1.1 or 2001:DB8::0DB8:800:200C:417A.
- **subnet** {*IPv4_address IPv4_mask* | *IPv6_address*/*IPv6_prefix*}—The address of a network. For IPv4 subnets, include the mask after a space, for example, 10.0.0.0 255.0.0.0. For IPv6, include the address and prefix as a single unit (no spaces), such as 2001:DB8:0:CD30::/60.

- **range** *start_address end_address*—A range of addresses. You can specify IPv4 or IPv6 ranges. Do not include masks or prefixes.

Example

```
hostname(config-network-object)# range 10.1.1.1 10.1.1.90
```

Step 4 Configure **dynamic PAT** for the object IP addresses. You can only define a single NAT rule for a given object.

```
nat [(real_ifc,mapped_ifc)] dynamic {mapped_inline_host_ip | mapped_obj |
pat-pool mapped_obj [round-robin] [extended] [flat [include-reserve]] | interface [ipv6]}
[interface [ipv6]] [dns]
```

Example

```
hostname(config-network-object)# nat (any,outside) dynamic interface
```

Where:

- Interfaces—(Required for transparent mode) Specify the real (*real_ifc*) and mapped (*mapped_ifc*) interfaces. Be sure to include the parentheses. In routed mode, if you do not specify the real and mapped interfaces, all interfaces are used. You can also specify the keyword **any** for one or both of the interfaces, for example (any,outside).
- Mapped IP address—You can specify the mapped IP address as:
  - *mapped_inline_host_ip*—An inline host address.
  - *mapped_obj*—An existing network object that is defined as a host address.
  - **pat-pool**—An existing network object or group that contains multiple addresses.
  - **interface**—(Routed mode only.) The IP address of the mapped interface is used as the mapped address. If you specify **ipv6**, then the IPv6 address of the interface is used. For this option, you must configure a specific interface for the *mapped_ifc*. You must use this keyword when you want to use the interface IP address; you cannot enter it inline or as an object.
- For a PAT pool, you can specify one or more of the following options:
  - Round robin—The **round-robin** keyword enables round-robin address allocation for a PAT pool. Without round robin, by default all ports for a PAT address will be allocated before the next PAT address is used. The round-robin method assigns an address/port from each PAT address in the pool before returning to use the first address again, and then the second address, and so on.
  - Extended PAT—The **extended** keyword enables extended PAT. Extended PAT uses 65535 ports per *service*, as opposed to per IP address, by including the destination address and port in the translation information. Normally, the destination port and address are not considered when creating PAT translations, so you are limited to 65535 ports per PAT address. For example, with extended PAT, you can create a translation of 10.1.1.1:1027 when going to 192.168.1.7:23 as well as a translation of 10.1.1.1:1027 when going to 192.168.1.7:80.
  - Flat range—The **flat** keyword enables use of the entire 1024 to 65535 port range when allocating ports. When choosing the mapped port number for a translation, the ASA uses the real source port number if it is available. However, without this option, if the real port is *not* available, by default the mapped ports are chosen from the same range of ports as the real port number: 1 to 511, 512 to 1023, and 1024 to 65535. To avoid running out of ports at the low ranges, configure this setting. To use the entire range of 1 to 65535, also specify the **include-reserve** keyword.

- Interface PAT fallback—(Optional.) The **interface** keyword enables interface PAT fallback when entered after a primary PAT address. After the primary PAT addresses are used up, then the IP address of the mapped interface is used. If you specify **ipv6**, then the IPv6 address of the interface is used. For this option, you must configure a specific interface for the *mapped_ifc*. (You cannot specify **interface** in transparent mode.)
- DNS—(Optional.) The **dns** keyword translates DNS replies. Be sure DNS inspection is enabled (it is enabled by default). See DNS and NAT, page 10-21 for more information.

Examples

The following example configures dynamic PAT that hides the 192.168.2.0 network behind address 10.2.2.2:

```
hostname(config)# object network my-inside-net
hostname(config-network-object)# subnet 192.168.2.0 255.255.255.0
hostname(config-network-object)# nat (inside,outside) dynamic 10.2.2.2
```

The following example configures dynamic PAT that hides the 192.168.2.0 network behind the outside interface address:

```
hostname(config)# object network my-inside-net
hostname(config-network-object)# subnet 192.168.2.0 255.255.255.0
hostname(config-network-object)# nat (inside,outside) dynamic interface
```

The following example configures dynamic PAT with a PAT pool to translate the inside IPv6 network to an outside IPv4 network:

```
hostname(config)# object network IPv4_POOL
hostname(config-network-object)# range 203.0.113.1 203.0.113.254
hostname(config)# object network IPv6_INSIDE
hostname(config-network-object)# subnet 2001:DB8::/96
hostname(config-network-object)# nat (inside,outside) dynamic pat-pool IPv4_POOL
```

Configure Dynamic Twice PAT

This section describes how to configure twice NAT for dynamic PAT.

Procedure

Step 1 Create host or range network objects (**object network** command), or network object groups (**object-group network** command), for the source real addresses, the source mapped addresses, the destination real addresses, and the destination mapped addresses.

- If you want to translate all source traffic, you can skip adding an object for the source real addresses, and instead specify the **any** keyword in the **nat** command.
- If you want to use the interface address as the mapped address, you can skip adding an object for the source mapped addresses, and instead specify the **interface** keyword in the **nat** command.
- If you want to configure destination static interface NAT with port translation only, you can skip adding an object for the destination mapped addresses, and instead specify the **interface** keyword in the **nat** command.

If you use an object, the object or group cannot contain a subnet. The object must define a host, or for a PAT pool, a range. The group (for a PAT pool) can include hosts and ranges.

Step 2 (Optional.) Create service objects for the destination real ports and the destination mapped ports.

For dynamic NAT, you can only perform port translation on the destination. A service object can contain both a source and destination port, but only the destination port is used in this case. If you specify the source port, it will be ignored.

Step 3 Configure **dynamic PAT**.

```
nat [(real_ifc,mapped_ifc)] [line | {after-auto [line]}]
source dynamic {real-obj | any}
{mapped_obj [interface [ipv6]] |
[pat-pool mapped_obj [round-robin] [extended] [flat [include-reserve]] [interface [ipv6]]
| interface [ipv6]}
[destination static {mapped_obj | interface [ipv6]} real_obj]
[service mapped_dest_svc_obj real_dest_svc_obj]
[dns] [unidirectional] [inactive] [description desc]
```

Example

```
hostname(config)# nat (inside,outside) source dynamic MyInsNet interface
destination static Server1 Server1
description Interface PAT for inside addresses when going to server 1
```

Where:

- Interfaces—(Required for transparent mode) Specify the real (*real_ifc*) and mapped (*mapped_ifc*) interfaces. Be sure to include the parentheses. In routed mode, if you do not specify the real and mapped interfaces, all interfaces are used. You can also specify the keyword **any** for one or both of the interfaces, for example (any,outside).
- Section and Line—(Optional.) By default, the NAT rule is added to the end of section 1 of the NAT table (see NAT Rule Order, page 9-5). If you want to add the rule into section 3 instead (after the network object NAT rules), then use the **after-auto** keyword. You can insert a rule anywhere in the applicable section using the *line* argument.
- Source addresses:
  - Real—Specify a network object, group, or the **any** keyword. Use the **any** keyword if you want to translate all traffic from the real interface to the mapped interface.
  - Mapped—Configure one of the following:

    - Network object—Specify a network object that contains a host address.

    - **pat-pool**—Specify the **pat-pool** keyword and a network object or group that contains multiple addresses.

    - **interface**—(Routed mode only.) Specify the **interface** keyword alone to only use interface PAT. If you specify **ipv6**, then the IPv6 address of the interface is used. When specified with a PAT pool or network object, the **interface** keyword enables interface PAT fallback. After the PAT IP addresses are used up, then the IP address of the mapped interface is used. For this option, you must configure a specific interface for the *mapped_ifc*.

    For a PAT pool, you can specify one or more of the following options:

    -- Round robin—The **round-robin** keyword enables round-robin address allocation for a PAT pool. Without round robin, by default all ports for a PAT address will be allocated before the next PAT address is used. The round-robin method assigns an address/port from each PAT address in the pool before returning to use the first address again, and then the second address, and so on.

    -- Extended PAT—The **extended** keyword enables extended PAT. Extended PAT uses 65535 ports per *service*, as opposed to per IP address, by including the destination address and port in the translation information. Normally, the destination port and address are not considered when

creating PAT translations, so you are limited to 65535 ports per PAT address. For example, with extended PAT, you can create a translation of 10.1.1.1:1027 when going to 192.168.1.7:23 as well as a translation of 10.1.1.1:1027 when going to 192.168.1.7:80.

-- Flat range—The **flat** keyword enables use of the entire 1024 to 65535 port range when allocating ports. When choosing the mapped port number for a translation, the ASA uses the real source port number if it is available. However, without this option, if the real port is *not* available, by default the mapped ports are chosen from the same range of ports as the real port number: 1 to 511, 512 to 1023, and 1024 to 65535. To avoid running out of ports at the low ranges, configure this setting. To use the entire range of 1 to 65535, also specify the **include-reserve** keyword.

- Destination addresses (Optional):
  - Mapped—Specify a network object or group, or for static interface NAT with port translation only (routed mode), specify the **interface** keyword. If you specify **ipv6**, then the IPv6 address of the interface is used. If you specify **interface**, be sure to also configure the **service** keyword. For this option, you must configure a specific interface for the *real_ifc*. See Static Interface NAT with Port Translation, page 9-29 for more information.
  - Real—Specify a network object or group. For identity NAT, simply use the same object or group for both the real and mapped addresses.
- Destination port—(Optional.) Specify the **service** keyword along with the mapped and real service objects. For identity port translation, simply use the same service object for both the real and mapped ports.
- DNS—(Optional; for a source-only rule.) The **dns** keyword translates DNS replies. Be sure DNS inspection is enabled (it is enabled by default). You cannot configure the **dns** keyword if you configure a **destination** address. See DNS and NAT, page 10-21 for more information.
- Unidirectional—(Optional.) Specify **unidirectional** so the destination addresses cannot initiate traffic to the source addresses.
- Inactive—(Optional.) To make this rule inactive without having to remove the command, use the **inactive** keyword. To reactivate it, reenter the whole command without the **inactive** keyword.
- Description—Optional.) Provide a description up to 200 characters using the **description** keyword.

Examples

The following example configures interface PAT for inside network 192.168.1.0/24 when accessing outside Telnet server 209.165.201.23, and Dynamic PAT using a PAT pool when accessing any server on the 203.0.113.0/24 network.

```
hostname(config)# object network INSIDE_NW
hostname(config-network-object)# subnet 192.168.1.0 255.255.255.0

hostname(config)# object network PAT_POOL
hostname(config-network-object)# range 209.165.200.225 209.165.200.254

hostname(config)# object network TELNET_SVR
hostname(config-network-object)# host 209.165.201.23

hostname(config)# object service TELNET
hostname(config-service-object)# service tcp destination eq 23

hostname(config)# object network SERVERS
hostname(config-network-object)# subnet 203.0.113.0 255.255.255.0
```

```
hostname(config)# nat (inside,outside) source dynamic INSIDE_NW interface
destination static TELNET_SVR TELNET_SVR service TELNET TELNET
hostname(config)# nat (inside,outside) source dynamic INSIDE_NW pat-pool PAT_POOL
destination static SERVERS SERVERS
```

The following example configures interface PAT for inside network 192.168.1.0/24 when accessing outside IPv6 Telnet server 2001:DB8::23, and Dynamic PAT using a PAT pool when accessing any server on the 2001:DB8:AAAA::/96 network.

```
hostname(config)# object network INSIDE_NW
hostname(config-network-object)# subnet 192.168.1.0 255.255.255.0

hostname(config)# object network PAT_POOL
hostname(config-network-object)# range 2001:DB8:AAAA::1 2001:DB8:AAAA::200

hostname(config)# object network TELNET_SVR
hostname(config-network-object)# host 2001:DB8::23

hostname(config)# object service TELNET
hostname(config-service-object)# service tcp destination eq 23

hostname(config)# object network SERVERS
hostname(config-network-object)# subnet 2001:DB8:AAAA::/96

hostname(config)# nat (inside,outside) source dynamic INSIDE_NW interface ipv6
destination static TELNET_SVR TELNET_SVR service TELNET TELNET
hostname(config)# nat (inside,outside) source dynamic INSIDE_NW pat-pool PAT_POOL
destination static SERVERS SERVERS
```

Configure Per-Session PAT or Multi-Session PAT

By default, all TCP PAT traffic and all UDP DNS traffic uses per-session PAT. To use multi-session PAT for traffic, you can configure per-session PAT rules: a permit rule uses per-session PAT, and a deny rule uses multi-session PAT.

Per-session PAT improves the scalability of PAT and, for clustering, allows each member unit to own PAT connections; multi-session PAT connections have to be forwarded to and owned by the master unit. At the end of a per-session PAT session, the ASA sends a reset and immediately removes the xlate. This reset causes the end node to immediately release the connection, avoiding the TIME_WAIT state. Multi-session PAT, on the other hand, uses the PAT timeout, by default 30 seconds.

For “hit-and-run” traffic, such as HTTP or HTTPS, per-session PAT can dramatically increase the connection rate supported by one address. Without per-session PAT, the maximum connection rate for one address for an IP protocol is approximately 2000 per second. With per-session PAT, the connection rate for one address for an IP protocol is 65535/*average-lifetime*.

For traffic that can benefit from multi-session PAT, such as H.323, SIP, or Skinny, you can disable per-session PAT by creating a per-session deny rule. These rules are available starting with version 9.0(1).

Before You Begin

By default, the following rules are installed:

```
xlate per-session permit tcp any4 any4
xlate per-session permit tcp any4 any6
xlate per-session permit tcp any6 any4
xlate per-session permit tcp any6 any6
xlate per-session permit udp any4 any4 eq domain
```

```
xlate per-session permit udp any4 any6 eq domain
xlate per-session permit udp any6 any4 eq domain
xlate per-session permit udp any6 any6 eq domain
```

You cannot remove these rules, and they always exist after any manually-created rules. Because rules are evaluated in order, you can override the default rules. For example, to completely negate these rules, you could add the following:

```
xlate per-session deny tcp any4 any4
xlate per-session deny tcp any4 any6
xlate per-session deny tcp any6 any4
xlate per-session deny tcp any6 any6
xlate per-session deny udp any4 any4 eq domain
xlate per-session deny udp any4 any6 eq domain
xlate per-session deny udp any6 any4 eq domain
xlate per-session deny udp any6 any6 eq domain
```

Procedure

Step 1 Create a permit or deny per-session PAT rule. This rule is placed above the default rules, but below any other manually-created rules. Be sure to create your rules in the order you want them applied.

xlate per-session {**permit** | **deny**} {**tcp** | **udp**} *source_ip* [*operator src_port*] *destination_ip* [*operator dest_port*]

Example

```
hostname(config)# xlate per-session deny tcp any4 209.165.201.3 eq 1720
```

For the source and destination IP addresses, you can configure the following:

- **host** *ip_address*—Specifies an IPv4 or IPv6 host address.
- *ip_address mask*—Specifies an IPv4 network address and subnet mask.
- *ipv6-address*/*prefix-length*—Specifies an IPv6 network address and prefix.
- **any4** and **any6**—**any4** specifies only IPv4 traffic; and **any6** specifies any6 traffic.

The *operator* matches the port numbers used by the source or destination. The default is all ports. The permitted operators are:

- **lt**—less than
- **gt**—greater than
- **eq**—equal to
- **neq**—not equal to
- **range**—an inclusive range of values. When you use this operator, specify two port numbers, for example, **range 100 200**.

Examples

The following example creates a deny rule for H.323 traffic, so that it uses multi-session PAT:

```
hostname(config)# xlate per-session deny tcp any4 209.165.201.7 eq 1720
hostname(config)# xlate per-session deny udp any4 209.165.201.7 range 1718 1719
```

Static NAT

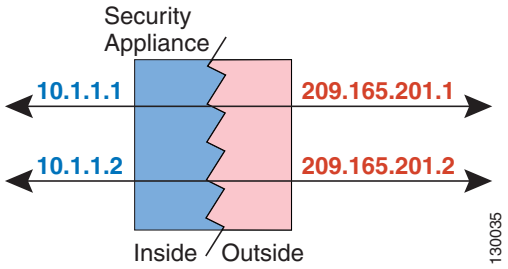
The following topics explain static NAT and how to implement it.

About Static NAT

Static NAT creates a fixed translation of a real address to a mapped address. Because the mapped address is the same for each consecutive connection, static NAT allows bidirectional connection initiation, both to and from the host (if an access rule exists that allows it). With dynamic NAT and PAT, on the other hand, each host uses a different address or port for each subsequent translation, so bidirectional initiation is not supported.

The following figure shows a typical static NAT scenario. The translation is always active so both real and remote hosts can initiate connections.

Figure 9-5 Static NAT



Note You can disable bidirectionality if desired.

Static NAT with Port Translation

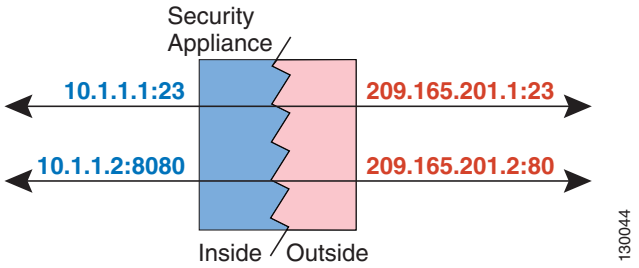
Static NAT with port translation lets you specify a real and mapped protocol (TCP or UDP) and port.

About Static NAT with Port Address Translation

When you specify the port with static NAT, you can choose to map the port and/or the IP address to the same value or to a different value.

The following figure shows a typical static NAT with port translation scenario showing both a port that is mapped to itself and a port that is mapped to a different value; the IP address is mapped to a different value in both cases. The translation is always active so both translated and remote hosts can initiate connections.

Figure 9-6 Typical Static NAT with Port Translation Scenario

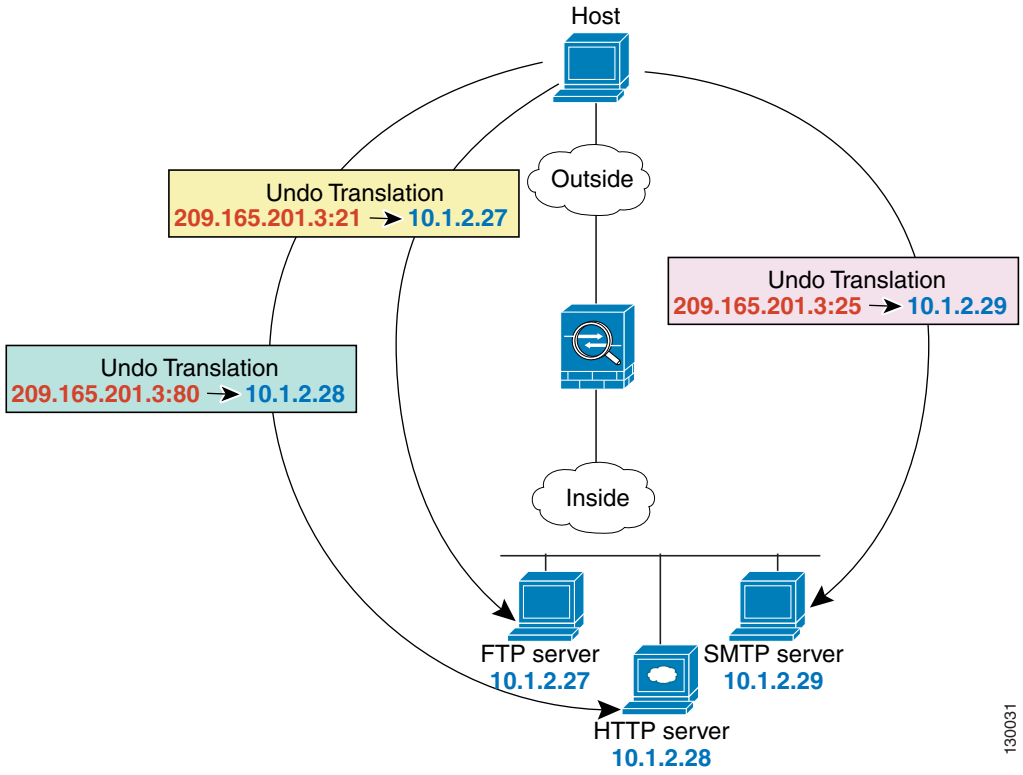


Note For applications that require application inspection for secondary channels (for example, FTP and VoIP), the ASA automatically translates the secondary ports.

Static NAT with Identity Port Translation

The following static NAT with port translation example provides a single address for remote users to access FTP, HTTP, and SMTP. These servers are actually different devices on the real network, but for each server, you can specify static NAT with port translation rules that use the same mapped IP address, but different ports. For details on how to configure this example, see Single Address for FTP, HTTP, and SMTP (Static NAT-with-Port-Translation), page 10-5.

Figure 9-7 Static NAT with Port Translation



Static NAT with Port Translation for Non-Standard Ports

You can also use static NAT with port translation to translate a well-known port to a non-standard port or vice versa. For example, if inside web servers use port 8080, you can allow outside users to connect to port 80, and then undo translation to the original port 8080. Similarly, to provide extra security, you can tell web users to connect to non-standard port 6785, and then undo translation to port 80.

Static Interface NAT with Port Translation

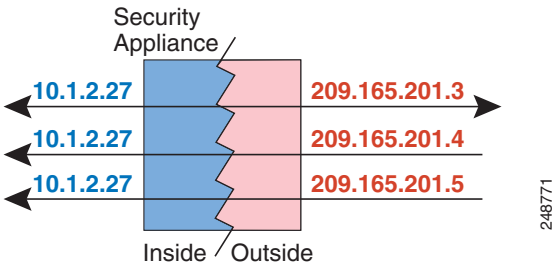
You can configure static NAT to map a real address to an interface address/port combination. For example, if you want to redirect Telnet access for the ASA outside interface to an inside host, then you can map the inside host IP address/port 23 to the ASA interface address/port 23. (Note that although Telnet to the ASA is not allowed to the lowest security interface, static NAT with interface port translation redirects the Telnet session instead of denying it).

One-to-Many Static NAT

Typically, you configure static NAT with a one-to-one mapping. However, in some cases, you might want to configure a single real address to several mapped addresses (one-to-many). When you configure one-to-many static NAT, when the real host initiates traffic, it always uses the first mapped address. However, for traffic initiated to the host, you can initiate traffic to any of the mapped addresses, and they will be untranslated to the single real address.

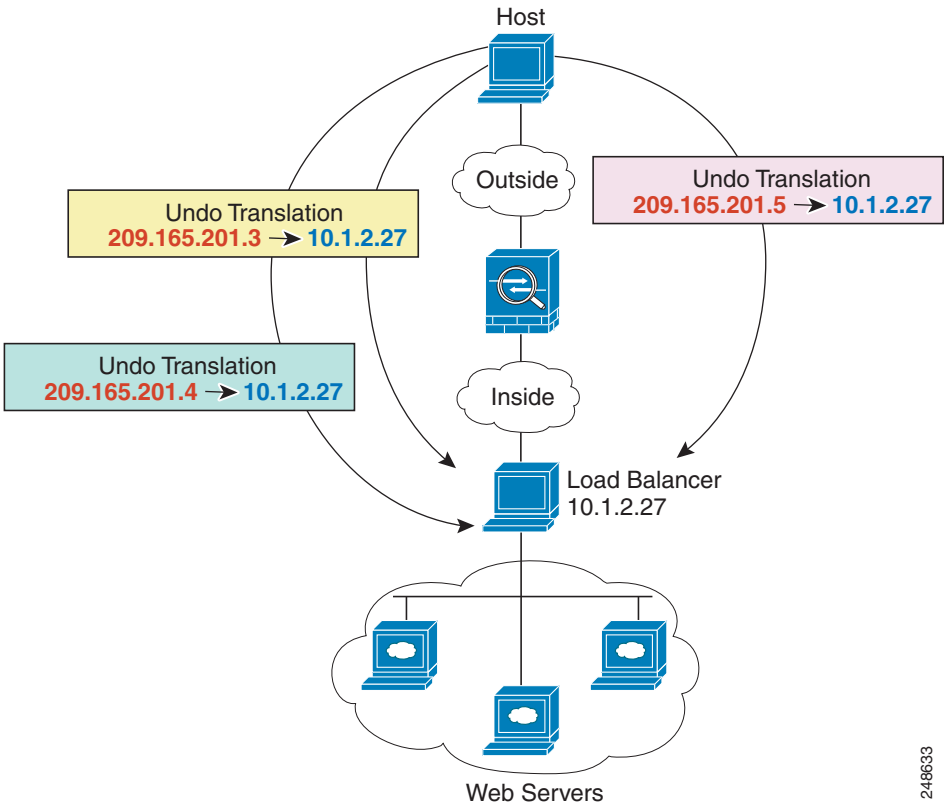
The following figure shows a typical one-to-many static NAT scenario. Because initiation by the real host always uses the first mapped address, the translation of real host IP/1st mapped IP is technically the only bidirectional translation.

Figure 9-8 One-to-Many Static NAT



For example, you have a load balancer at 10.1.2.27. Depending on the URL requested, it redirects traffic to the correct web server. For details on how to configure this example, see Inside Load Balancer with Multiple Mapped Addresses (Static NAT, One-to-Many), page 10-4.

Figure 9-9 One-to-Many Static NAT Example



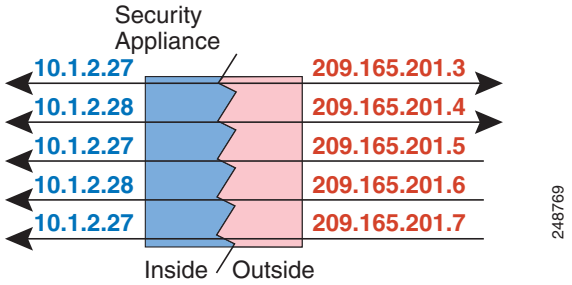
Other Mapping Scenarios (Not Recommended)

The ASA has the flexibility to allow any kind of static mapping scenario: one-to-one, one-to-many, but also few-to-many, many-to-few, and many-to-one mappings. We recommend using only one-to-one or one-to-many mappings. These other mapping options might result in unintended consequences.

Functionally, few-to-many is the same as one-to-many; but because the configuration is more complicated and the actual mappings may not be obvious at a glance, we recommend creating a one-to-many configuration for each real address that requires it. For example, for a few-to-many scenario, the few real addresses are mapped to the many mapped addresses in order (A to 1, B to 2, C to 3). When all real addresses are mapped, the next mapped address is mapped to the first real address, and so on until all mapped addresses are mapped (A to 4, B to 5, C to 6). This results in multiple mapped addresses for each real address. Just like a one-to-many configuration, only the first mappings are bidirectional; subsequent mappings allow traffic to be initiated *to* the real host, but all traffic *from* the real host uses only the first mapped address for the source.

The following figure shows a typical few-to-many static NAT scenario.

Figure 9-10 Few-to-Many Static NAT

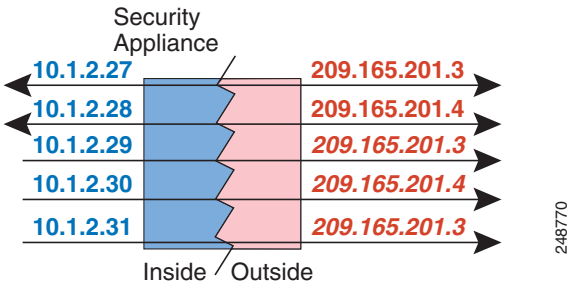


For a many-to-few or many-to-one configuration, where you have more real addresses than mapped addresses, you run out of mapped addresses before you run out of real addresses. Only the mappings between the lowest real IP addresses and the mapped pool result in bidirectional initiation. The remaining higher real addresses can initiate traffic, but traffic cannot be initiated to them (returning traffic for a connection is directed to the correct real address because of the unique 5-tuple (source IP, destination IP, source port, destination port, protocol) for the connection).

Note Many-to-few or many-to-one NAT is not PAT. If two real hosts use the same source port number and go to the same outside server and the same TCP destination port, and both hosts are translated to the same IP address, then both connections will be reset because of an address conflict (the 5-tuple is not unique).

The following figure shows a typical many-to-few static NAT scenario.

Figure 9-11 Many-to-Few Static NAT



Instead of using a static rule this way, we suggest that you create a one-to-one rule for the traffic that needs bidirectional initiation, and then create a dynamic rule for the rest of your addresses.

Configure Static Network Object NAT or Static NAT-with-Port-Translation

This section describes how to configure a static NAT rule using network object NAT.

Procedure

Step 1 (Optional.) Create a network object (**object network** command), or a network object group (**object-group network** command), for the mapped addresses.

- Instead of using an object, you can configure an inline address or specify the interface address (for static NAT-with-port-translation).
- If you use an object, the object or group can contain a host, range, or subnet.

Step 2 Create or edit the network object for which you want to configure NAT.

```
object network obj_name
```

Example

```
hostname(config)# object network my-host-obj1
```

Step 3 (Skip when editing an object that has the right address.) Define the real IPv4 or IPv6 addresses that you want to translate.

- **host** {*IPv4_address* | *IPv6_address*}—The IPv4 or IPv6 address of a single host. For example, 10.1.1.1 or 2001:DB8::0DB8:800:200C:417A.
- **subnet** {*IPv4_address IPv4_mask* | *IPv6_address*/*IPv6_prefix*}—The address of a network. For IPv4 subnets, include the mask after a space, for example, 10.0.0.0 255.0.0.0. For IPv6, include the address and prefix as a single unit (no spaces), such as 2001:DB8:0:CD30::/60.
- **range** *start_address end_address*—A range of addresses. You can specify IPv4 or IPv6 ranges. Do not include masks or prefixes.

Example

```
hostname(config-network-object)# subnet 10.2.1.0 255.255.255.0
```

Step 4 Configure **static NAT** for the object IP addresses. You can only define a single NAT rule for a given object.

```
nat [(real_ifc,mapped_ifc)] static {mapped_inline_ip | mapped_obj | interface [ipv6]}
[net-to-net] [dns | service {tcp | udp} real_port mapped_port] [no-proxy-arp]
```

Example

```
hostname(config-network-object)#
nat (inside,outside) static MAPPED_IPS service tcp 80 8080
```

Where:

- Interfaces—(Required for transparent mode) Specify the real (*real_ifc*) and mapped (*mapped_ifc*) interfaces. Be sure to include the parentheses. In routed mode, if you do not specify the real and mapped interfaces, all interfaces are used. You can also specify the keyword **any** for one or both of the interfaces, for example (any,outside).
- Mapped IP address—You can specify the mapped IP address as one of the following. Typically, you configure the same number of mapped addresses as real addresses for a one-to-one mapping. You can, however, have a mismatched number of addresses. See Static NAT, page 9-27.
  - *mapped_inline_host_ip*—An inline IP address. The netmask, prefix, or range for the mapped network is the same as that of the real network. For example, if the real network is a host, then this address will be a host address. In the case of a range, then the mapped addresses include the same number of addresses as the real range. For example, if the real address is defined as a range from 10.1.1.1 through 10.1.1.6, and you specify 172.20.1.1 as the mapped address, then the mapped range will include 172.20.1.1 through 172.20.1.6.
  - *mapped_obj*—An existing network object or group.
  - **interface**—(Static NAT-with-port-translation only; routed mode only.) The IP address of the mapped interface is used as the mapped address. If you specify **ipv6**, then the IPv6 address of the interface is used. For this option, you must configure a specific interface for the *mapped_ifc*. You must use this keyword when you want to use the interface IP address; you cannot enter it inline or as an object. Be sure to also configure the **service** keyword.
- Net-to-net—(Optional.) For NAT 46, specify **net-to-net** to translate the first IPv4 address to the first IPv6 address, the second to the second, and so on. Without this option, the IPv4-embedded method is used. For a one-to-one translation, you must use this keyword.
- DNS—(Optional.) The **dns** keyword translates DNS replies. Be sure DNS inspection is enabled (it is enabled by default). See DNS and NAT, page 10-21 for more information.
- Port translation—(Static NAT-with-port-translation only.) Specify **service** with either **tcp** or **udp** and the real and mapped ports. You can enter either a port number or a well-known port name (such as **ftp**).
- No Proxy ARP—(Optional.) Specify **no-proxy-arp** to disable proxy ARP for incoming packets to the mapped IP addresses. For information on the conditions which might require the disabling of proxy ARP, see Mapped Addresses and Routing, page 10-12.

Examples

The following example configures static NAT for the real host 10.1.1.1 on the inside to 10.2.2.2 on the outside with DNS rewrite enabled.

```
hostname(config)# object network my-host-obj1
hostname(config-network-object)# host 10.1.1.1
hostname(config-network-object)# nat (inside,outside) static 10.2.2.2 dns
```

The following example configures static NAT for the real host 10.1.1.1 on the inside to 10.2.2.2 on the outside using a mapped object.

```
hostname(config)# object network my-mapped-obj
hostname(config-network-object)# host 10.2.2.2

hostname(config-network-object)# object network my-host-obj1
hostname(config-network-object)# host 10.1.1.1
hostname(config-network-object)# nat (inside,outside) static my-mapped-obj
```

The following example configures static NAT-with-port-translation for 10.1.1.1 at TCP port 21 to the outside interface at port 2121.

```
hostname(config)# object network my-ftp-server
hostname(config-network-object)# host 10.1.1.1
hostname(config-network-object)# nat (inside,outside) static interface service tcp 21 2121
```

The following example maps an inside IPv4 network to an outside IPv6 network.

```
hostname(config)# object network inside_v4_v6
hostname(config-network-object)# subnet 10.1.1.0 255.255.255.0
hostname(config-network-object)# nat (inside,outside) static 2001:DB8::/96
```

The following example maps an inside IPv6 network to an outside IPv6 network.

```
hostname(config)# object network inside_v6
hostname(config-network-object)# subnet 2001:DB8:AAAA::/96
hostname(config-network-object)# nat (inside,outside) static 2001:DB8:BBBB::/96
```

Configure Static Twice NAT or Static NAT-with-Port-Translation

This section describes how to configure a static NAT rule using twice NAT.

Procedure

Step 1 Create host or range network objects (**object network** command), or network object groups (**object-group network** command), for the source real addresses, the source mapped addresses, the destination real addresses, and the destination mapped addresses.

- If you want to configure source static interface NAT with port translation only, you can skip adding an object for the source mapped addresses, and instead specify the **interface** keyword in the **nat** command.
- If you want to configure destination static interface NAT with port translation only, you can skip adding an object for the destination mapped addresses, and instead specify the **interface** keyword in the **nat** command.

If you do create objects, consider the following guidelines:

- The mapped object or group can contain a host, range, or subnet.
- The static mapping is typically one-to-one, so the real addresses have the same quantity as the mapped addresses. You can, however, have different quantities if desired. For more information, see Static NAT, page 9-27.

Step 2 (Optional.) Create service objects for the:

- Source *or* Destination real ports
- Source *or* Destination mapped ports

A service object can contain both a source and destination port; however, you should specify *either* the source *or* the destination port for both service objects. You should only specify *both* the source and destination ports if your application uses a fixed source port (such as some DNS servers); but fixed source ports are rare. For example, if you want to translate the port for the source host, then configure the source service.

Step 3 Configure **static NAT**.

```
nat [(real_ifc,mapped_ifc)] [line | {after-object [line]}]
source static real_ob [mapped_obj | interface [ipv6]]
[destination static {mapped_obj | interface [ipv6]} real_obj]
[service real_src_mapped_dest_svc_obj mapped_src_real_dest_svc_obj]
[net-to-net] [dns] [unidirectional | no-proxy-arp] [inactive] [description desc]
```

Example

```
hostname(config)# nat (inside,dmz) source static MyInsNet MyInsNet_mapped
destination static Server1 Server1 service REAL_SRC_SVC MAPPED_SRC_SVC
```

Where:

- Interfaces—(Required for transparent mode) Specify the real (*real_ifc*) and mapped (*mapped_ifc*) interfaces. Be sure to include the parentheses. In routed mode, if you do not specify the real and mapped interfaces, all interfaces are used. You can also specify the keyword **any** for one or both of the interfaces, for example (any,outside).
- Section and Line—(Optional.) By default, the NAT rule is added to the end of section 1 of the NAT table (see NAT Rule Order, page 9-5). If you want to add the rule into section 3 instead (after the network object NAT rules), then use the **after-auto** keyword. You can insert a rule anywhere in the applicable section using the *line* argument.
- Source addresses:
  - Real—Specify a network object or group. Do not use the **any** keyword, which would be used for identity NAT.
  - Mapped—Specify a different network object or group. For static interface NAT with port translation only, you can specify the **interface** keyword (routed mode only). If you specify **ipv6**, then the IPv6 address of the interface is used. If you specify **interface**, be sure to also configure the **service** keyword (in this case, the service objects should include only the source port). For this option, you must configure a specific interface for the *mapped_ifc*. See Static Interface NAT with Port Translation, page 9-29 for more information.
- Destination addresses (Optional):
  - Mapped—Specify a network object or group, or for static interface NAT with port translation only, specify the **interface** keyword. If you specify **ipv6**, then the IPv6 address of the interface is used. If you specify **interface**, be sure to also configure the **service** keyword (in this case, the service objects should include only the destination port). For this option, you must configure a specific interface for the *real_ifc*.
  - Real—Specify a network object or group. For identity NAT, simply use the same object or group for both the real and mapped addresses.
- Ports—(Optional.) Specify the **service** keyword along with the real and mapped service objects. For source port translation, the objects must specify the source service. The order of the service objects in the command for source port translation is **service** *real_obj mapped_obj*. For destination port translation, the objects must specify the destination service. The order of the service objects for destination port translation is **service** *mapped_obj real_obj*. In the rare case where you specify both the source and destination ports in the object, the first service object contains the real source

port/mapped destination port; the second service object contains the mapped source port/real destination port. For identity port translation, simply use the same service object for both the real and mapped ports (source and/or destination ports, depending on your configuration).

- Net-to-net—(Optional.) For NAT 46, specify **net-to-net** to translate the first IPv4 address to the first IPv6 address, the second to the second, and so on. Without this option, the IPv4-embedded method is used. For a one-to-one translation, you must use this keyword.
- DNS—(Optional; for a source-only rule.) The **dns** keyword translates DNS replies. Be sure DNS inspection is enabled (it is enabled by default). You cannot configure the **dns** keyword if you configure a **destination** address. See DNS and NAT, page 10-21 for more information.
- Unidirectional—(Optional.) Specify **unidirectional** so the destination addresses cannot initiate traffic to the source addresses.
- No Proxy ARP—(Optional.) Specify **no-proxy-arp** to disable proxy ARP for incoming packets to the mapped IP addresses. See Mapped Addresses and Routing, page 10-12 for more information.
- Inactive—(Optional.) To make this rule inactive without having to remove the command, use the **inactive** keyword. To reactivate it, reenter the whole command without the **inactive** keyword.
- Description—Optional.) Provide a description up to 200 characters using the **description** keyword.

Examples

The following example shows the use of static interface NAT with port translation. Hosts on the outside access an FTP server on the inside by connecting to the outside interface IP address with destination port 65000 through 65004. The traffic is untranslated to the internal FTP server at 192.168.10.100:6500 through 65004. Note that you specify the source port range in the service object (and not the destination port) because you want to translate the source address and port as identified in the command; the destination port is “any.” Because static NAT is bidirectional, “source” and “destination” refers primarily to the command keywords; the actual source and destination address and port in a packet depends on which host sent the packet. In this example, connections are originated from outside to inside, so the “source” address and port of the FTP server is actually the destination address and port in the originating packet.

```
hostname(config)# object service FTP_PASV_PORT_RANGE
hostname(config-service-object)# service tcp source range 65000 65004

hostname(config)# object network HOST_FTP_SERVER
hostname(config-network-object)# host 192.168.10.100

hostname(config)# nat (inside,outside) source static HOST_FTP_SERVER interface
service FTP_PASV_PORT_RANGE FTP_PASV_PORT_RANGE
```

The following example shows a static translation of one IPv6 network to another IPv6 when accessing an IPv6 network, and the dynamic PAT translation to an IPv4 PAT pool when accessing the IPv4 network:

```
hostname(config)# object network INSIDE_NW
hostname(config-network-object)# subnet 2001:DB8:AAAA::/96

hostname(config)# object network MAPPED_IPv6_NW
hostname(config-network-object)# subnet 2001:DB8:BBBB::/96

hostname(config)# object network OUTSIDE_IPv6_NW
hostname(config-network-object)# subnet 2001:DB8:CCCC::/96

hostname(config)# object network OUTSIDE_IPv4_NW
hostname(config-network-object)# subnet 10.1.1.0 255.255.255.0
```

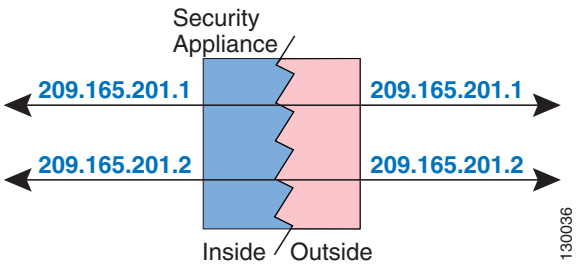
```
hostname(config)# object network MAPPED_IPv4_POOL
hostname(config-network-object)# range 10.1.2.1 10.1.2.254

hostname(config)# nat (inside,outside) source static INSIDE_NW MAPPED_IPv6_NW
destination static OUTSIDE_IPv6_NW OUTSIDE_IPv6_NW
hostname(config)# nat (inside,outside) source dynamic INSIDE_NW pat-pool MAPPED_IPv4_POOL
destination static OUTSIDE_IPv4_NW OUTSIDE_IPv4_NW
```

Identity NAT

You might have a NAT configuration in which you need to translate an IP address to itself. For example, if you create a broad rule that applies NAT to every network, but want to exclude one network from NAT, you can create a static NAT rule to translate an address to itself. Identity NAT is necessary for remote access VPN, where you need to exempt the client traffic from NAT.

The following figure shows a typical identity NAT scenario.

Figure 9-12 Identity NAT



The following topics explain how to configure identity NAT.

- Configure Identity Network Object NAT, page 9-37
- Configure Identity Twice NAT, page 9-39

Configure Identity Network Object NAT

This section describes how to configure an identity NAT rule using network object NAT.

Procedure

Step 1 (Optional.) Create a network object (**object network** command), or a network object group (**object-group network** command), for the mapped addresses.

- Instead of using an object, you can configure an inline address.
- If you use an object, the object must match the real addresses you want to translate.

Step 2 Create or edit the network object for which you want to configure NAT. The object must be a different one than what you use for the mapped addresses, even though the contents must be the same in each object.

object network *obj_name*

Example

```
hostname(config)# object network my-host-obj1
```

Step 3 (Skip when editing an object that has the right address.) Define the real IPv4 or IPv6 addresses that you want to translate.

- **host** {*IPv4_address* | *IPv6_address*}—The IPv4 or IPv6 address of a single host. For example, 10.1.1.1 or 2001:DB8::0DB8:800:200C:417A.
- **subnet** {*IPv4_address IPv4_mask* | *IPv6_address*/*IPv6_prefix*}—The address of a network. For IPv4 subnets, include the mask after a space, for example, 10.0.0.0 255.0.0.0. For IPv6, include the address and prefix as a single unit (no spaces), such as 2001:DB8:0:CD30::/60.
- **range** *start_address end_address*—A range of addresses. You can specify IPv4 or IPv6 ranges. Do not include masks or prefixes.

Example

```
hostname(config-network-object)# subnet 10.2.1.0 255.255.255.0
```

Step 4 Configure **identity NAT** for the object IP addresses. You can only define a single NAT rule for a given object.

nat [**(***real_ifc***,***mapped_ifc***)**] **static** {*mapped_inline_ip* | *mapped_obj*} [**no-proxy-arp**] [**route-lookup**]

Example

```
hostname(config-network-object)# nat (inside,outside) static MAPPED_IPS
```

Where:

- Interfaces—(Required for transparent mode) Specify the real (*real_ifc*) and mapped (*mapped_ifc*) interfaces. Be sure to include the parentheses. In routed mode, if you do not specify the real and mapped interfaces, all interfaces are used. You can also specify the keyword **any** for one or both of the interfaces, for example (any,outside).
- Mapped IP addresses—Be sure to configure the same IP address for both the mapped and real address. Use one of the following:
  - *mapped_inline_host_ip*—An inline IP address. The netmask, prefix, or range for the mapped network is the same as that of the real network. For example, if the real network is a host, then this address will be a host address. In the case of a range, then the mapped addresses include the same number of addresses as the real range. For example, if the real address is defined as a range from 10.1.1.1 through 10.1.1.6, and you specify 10.1.1.1 as the mapped address, then the mapped range will include 10.1.1.1 through 10.1.1.6.
  - *mapped_obj*—A network object or group that includes the same addresses as the real object.
- No Proxy ARP—(Optional.) Specify **no-proxy-arp** to disable proxy ARP for incoming packets to the mapped IP addresses. For information on the conditions which might require the disabling of proxy ARP, see Mapped Addresses and Routing, page 10-12.
- Route lookup—(Routed mode only; interfaces specified.) Specify **route-lookup** to determine the egress interface using a route lookup instead of using the interface specified in the NAT command. See Determining the Egress Interface, page 10-14 for more information.

Example

The following example maps a host address to itself using an inline mapped address:

```
hostname(config)# object network my-host-obj1
hostname(config-network-object)# host 10.1.1.1
hostname(config-network-object)# nat (inside,outside) static 10.1.1.1
```

The following example maps a host address to itself using a network object:

```
hostname(config)# object network my-host-obj1-identity
hostname(config-network-object)# host 10.1.1.1

hostname(config-network-object)# object network my-host-obj1
hostname(config-network-object)# host 10.1.1.1
hostname(config-network-object)# nat (inside,outside) static my-host-obj1-identity
```

Configure Identity Twice NAT

This section describes how to configure an identity NAT rule using twice NAT.

Procedure

Step 1 Create host or range network objects (**object network** command), or network object groups (**object-group network** command), for the source real addresses (you will typically use the same object for the source mapped addresses), the destination real addresses, and the destination mapped addresses.

- If you want to perform identity NAT for all addresses, you can skip creating an object for the source real addresses and instead use the keywords **any any** in the **nat** command.
- If you want to configure destination static interface NAT with port translation only, you can skip adding an object for the destination mapped addresses, and instead specify the **interface** keyword in the **nat** command.

If you do create objects, consider the following guidelines:

- The mapped object or group can contain a host, range, or subnet.
- The real and mapped source objects must match. You can use the same object for both, or you can create separate objects that contain the same IP addresses.

Step 2 (Optional.) Create service objects for the:

- Source *or* Destination real ports
- Source *or* Destination mapped ports

A service object can contain both a source and destination port; however, you should specify *either* the source *or* the destination port for both service objects. You should only specify *both* the source and destination ports if your application uses a fixed source port (such as some DNS servers); but fixed source ports are rare. For example, if you want to translate the port for the source host, then configure the source service.

Step 3 Configure **identity NAT**.

```
nat [(real_ifc,mapped_ifc)] [line | {after-object [line]}]
source static {nw_obj nw_obj | any any}
[destination static {mapped_obj | interface [ipv6]} real_obj]
[service real_src_mapped_dest_svc_obj mapped_src_real_dest_svc_obj]
[no-proxy-arp] [route-lookup] [inactive] [description desc]
```

Example

```
hostname(config)# nat (inside,outside) source static MyInsNet MyInsNet
destination static Server1 Server1
```

Where:

- Interfaces—(Required for transparent mode.) Specify the real (*real_ifc*) and mapped (*mapped_ifc*) interfaces. Be sure to include the parentheses. In routed mode, if you do not specify the real and mapped interfaces, all interfaces are used. You can also specify the keyword **any** for one or both of the interfaces, for example (any,outside).
- Section and Line—(Optional.) By default, the NAT rule is added to the end of section 1 of the NAT table (see NAT Rule Order, page 9-5). If you want to add the rule into section 3 instead (after the network object NAT rules), then use the **after-auto** keyword. You can insert a rule anywhere in the applicable section using the *line* argument.
- Source addresses—Specify a network object, group, or the **any** keyword for both the real and mapped addresses.
- Destination addresses (Optional):
  - Mapped—Specify a network object or group, or for static interface NAT with port translation only, specify the **interface** keyword (routed mode only). If you specify **ipv6**, then the IPv6 address of the interface is used. If you specify **interface**, be sure to also configure the **service** keyword (in this case, the service objects should include only the destination port). For this option, you must configure a specific interface for the *real_ifc*.
  - Real—Specify a network object or group. For identity NAT, simply use the same object or group for both the real and mapped addresses.
- Ports—(Optional.) Specify the **service** keyword along with the real and mapped service objects. For source port translation, the objects must specify the source service. The order of the service objects in the command for source port translation is **service** *real_obj mapped_obj*. For destination port translation, the objects must specify the destination service. The order of the service objects for destination port translation is **service** *mapped_obj real_obj*. In the rare case where you specify both the source and destination ports in the object, the first service object contains the real source port/mapped destination port; the second service object contains the mapped source port/real destination port. For identity port translation, simply use the same service object for both the real and mapped ports (source and/or destination ports, depending on your configuration).
- No Proxy ARP—(Optional.) Specify **no-proxy-arp** to disable proxy ARP for incoming packets to the mapped IP addresses. See Mapped Addresses and Routing, page 10-12 for more information.
- Route lookup—(Optional; routed mode only; interfaces specified.) Specify **route-lookup** to determine the egress interface using a route lookup instead of using the interface specified in the NAT command. See Determining the Egress Interface, page 10-14 for more information.
- Inactive—(Optional.) To make this rule inactive without having to remove the command, use the **inactive** keyword. To reactivate it, reenter the whole command without the **inactive** keyword.
- Description—Optional.) Provide a description up to 200 characters using the **description** keyword.

Monitoring NAT

To monitor object NAT, use the following commands:

- **show nat**

Shows NAT statistics, including hits for each NAT rule.

- **show nat pool**

  Shows NAT pool statistics, including the addresses and ports allocated, and how many times they were allocated.

- **show running-config nat**

  Shows the NAT configuration. You cannot see object NAT rules using **show running-config object**. When you use show running-config without modifiers, objects that include NAT rules are shown twice, first with the basic address configuration, then later in the configuration, the object with the NAT rule. The complete object, with the address and NAT rule, is not shown as a unit.

- **show xlate**

  Shows current NAT session information.

History for NAT

Feature Name	Platform Releases	Description
Network Object NAT	8.3(1)	Configures NAT for a network object IP address(es). We introduced or modified the following commands: **nat** (object network configuration mode), **show nat**, **show xlate**, **show nat pool**.
Twice NAT	8.3(1)	Twice NAT lets you identify both the source and destination address in a single rule. We modified or introduced the following commands: **nat**, **show nat**, **show xlate**, **show nat pool**.

Feature Name	Platform Releases	Description
Identity NAT configurable proxy ARP and route lookup	8.4(2)/8.5(1)	In earlier releases for identity NAT, proxy ARP was disabled, and a route lookup was always used to determine the egress interface. You could not configure these settings. In 8.4(2) and later, the default behavior for identity NAT was changed to match the behavior of other static NAT configurations: proxy ARP is enabled, and the NAT configuration determines the egress interface (if specified) by default. You can leave these settings as is, or you can enable or disable them discretely. Note that you can now also disable proxy ARP for regular static NAT. For pre-8.3 configurations, the migration of NAT exempt rules (the **nat 0 access-list** command) to 8.4(2) and later now includes the following keywords to disable proxy ARP and to use a route lookup: **no-proxy-arp** and **route-lookup**. The **unidirectional** keyword that was used for migrating to 8.3(2) and 8.4(1) is no longer used for migration. When upgrading to 8.4(2) from 8.3(1), 8.3(2), and 8.4(1), all identity NAT configurations will now include the **no-proxy-arp** and **route-lookup** keywords, to maintain existing functionality. The **unidirectional** keyword is removed. We modified the following command: **nat static** [**no-proxy-arp**] [**route-lookup**].
PAT pool and round robin address assignment	8.4(2)/8.5(1)	You can now specify a pool of PAT addresses instead of a single address. You can also optionally enable round-robin assignment of PAT addresses instead of first using all ports on a PAT address before using the next address in the pool. These features help prevent a large number of connections from a single PAT address from appearing to be part of a DoS attack and makes configuration of large numbers of PAT addresses easy. We modifed the following commands: **nat dynamic** [**pat-pool** *mapped_object* [**round-robin**]] and **nat source dynamic** [**pat-pool** *mapped_object* [**round-robin**]].
Round robin PAT pool allocation uses the same IP address for existing hosts	8.4(3)	When using a PAT pool with round robin allocation, if a host has an existing connection, then subsequent connections from that host will use the same PAT IP address if ports are available. We did not modify any commands. *This feature is not available in 8.5(1) or 8.6(1).*

Feature Name	Platform Releases	Description
Flat range of PAT ports for a PAT pool	8.4(3)	If available, the real source port number is used for the mapped port. However, if the real port is *not* available, by default the mapped ports are chosen from the same range of ports as the real port number: 0 to 511, 512 to 1023, and 1024 to 65535. Therefore, ports below 1024 have only a small PAT pool. If you have a lot of traffic that uses the lower port ranges, when using a PAT pool, you can now specify a flat range of ports to be used instead of the three unequal-sized tiers: either 1024 to 65535, or 1 to 65535. We modifed the following commands: **nat dynamic** [**pat-pool** *mapped_object* [**flat** [**include-reserve**]]] and **nat source dynamic** [**pat-pool** *mapped_object* [**flat** [**include-reserve**]]]. *This feature is not available in 8.5(1) or 8.6(1).*
Extended PAT for a PAT pool	8.4(3)	Each PAT IP address allows up to 65535 ports. If 65535 ports do not provide enough translations, you can now enable extended PAT for a PAT pool. Extended PAT uses 65535 ports per *service*, as opposed to per IP address, by including the destination address and port in the translation information. We modifed the following command: **nat dynamic** [**pat-pool** *mapped_object* [**extended**]] and **nat source dynamic** [**pat-pool** *mapped_object* [**extended**]]. *This feature is not available in 8.5(1) or 8.6(1).*

Feature Name	Platform Releases	Description
Automatic NAT rules to translate a VPN peer's local IP address back to the peer's real IP address	8.4(3)	In rare situations, you might want to use a VPN peer's real IP address on the inside network instead of an assigned local IP address. Normally with VPN, the peer is given an assigned local IP address to access the inside network. However, you might want to translate the local IP address back to the peer's real public IP address if, for example, your inside servers and network security is based on the peer's real IP address. You can enable this feature on one interface per tunnel group. Object NAT rules are dynamically added and deleted when the VPN session is established or disconnected. You can view the rules using the **show nat** command. Because of routing issues, we do not recommend using this feature unless you know you need it; contact Cisco TAC to confirm feature compatibility with your network. See the following limitations: • Only supports Cisco IPsec and AnyConnect Client. • Return traffic to the public IP addresses must be routed back to the ASA so the NAT policy and VPN policy can be applied. • Does not support load-balancing (because of routing issues). • Does not support roaming (public IP changing). We introduced the following command: **nat-assigned-to-public-ip** *interface* (tunnel-group general-attributes configuration mode).
NAT support for IPv6	9.0(1)	NAT now supports IPv6 traffic, as well as translating between IPv4 and IPv6. Translating between IPv4 and IPv6 is not supported in transparent mode. We modified the following commands: **nat** (global and object network configuration modes), **show nat**, **show nat pool**, **show xlate**.
NAT support for reverse DNS lookups	9.0(1)	NAT now supports translation of the DNS PTR record for reverse DNS lookups when using IPv4 NAT, IPv6 NAT, and NAT64 with DNS inspection enabled for the NAT rule.

Feature Name	Platform Releases	Description
Per-session PAT	9.0(1)	The per-session PAT feature improves the scalability of PAT and, for clustering, allows each member unit to own PAT connections; multi-session PAT connections have to be forwarded to and owned by the master unit. At the end of a per-session PAT session, the ASA sends a reset and immediately removes the xlate. This reset causes the end node to immediately release the connection, avoiding the TIME_WAIT state. Multi-session PAT, on the other hand, uses the PAT timeout, by default 30 seconds. For "hit-and-run" traffic, such as HTTP or HTTPS, the per-session feature can dramatically increase the connection rate supported by one address. Without the per-session feature, the maximum connection rate for one address for an IP protocol is approximately 2000 per second. With the per-session feature, the connection rate for one address for an IP protocol is 65535/*average-lifetime*. By default, all TCP traffic and UDP DNS traffic use a per-session PAT xlate. For traffic that requires multi-session PAT, such as H.323, SIP, or Skinny, you can disable per-session PAT by creating a per-session deny rule. We introduced the following commands: **xlate per-session**, **show nat pool**.
Transactional Commit Model on NAT Rule Engine	9.3(1)	When enabled, a NAT rule update is applied after the rule compilation is completed; without affecting the rule matching performance. We added the **nat** keyword to the following commands: **asp rule-engine transactional-commit**, **show running-config asp rule-engine transactional-commit**, **clear configure asp rule-engine transactional-commit**. to

NAT Examples and Reference

The following topics provide examples for configuring NAT, plus information on advanced configuration and troubleshooting.

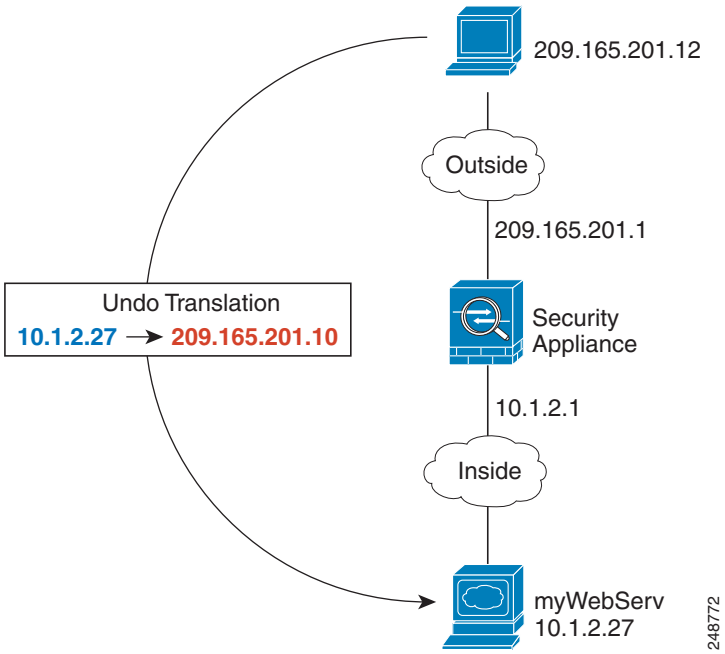
Examples for Network Object NAT

Following are some configuration examples for network object NAT.

Providing Access to an Inside Web Server (Static NAT)

The following example performs static NAT for an inside web server. The real address is on a private network, so a public address is required. Static NAT is necessary so hosts can initiate traffic to the web server at a fixed address.

Figure 10-1 Static NAT for an Inside Web Server



Procedure

Step 1 Create a network object for the internal web server.

```
hostname(config)# object network myWebServ
hostname(config-network-object)# host 10.1.2.27
```

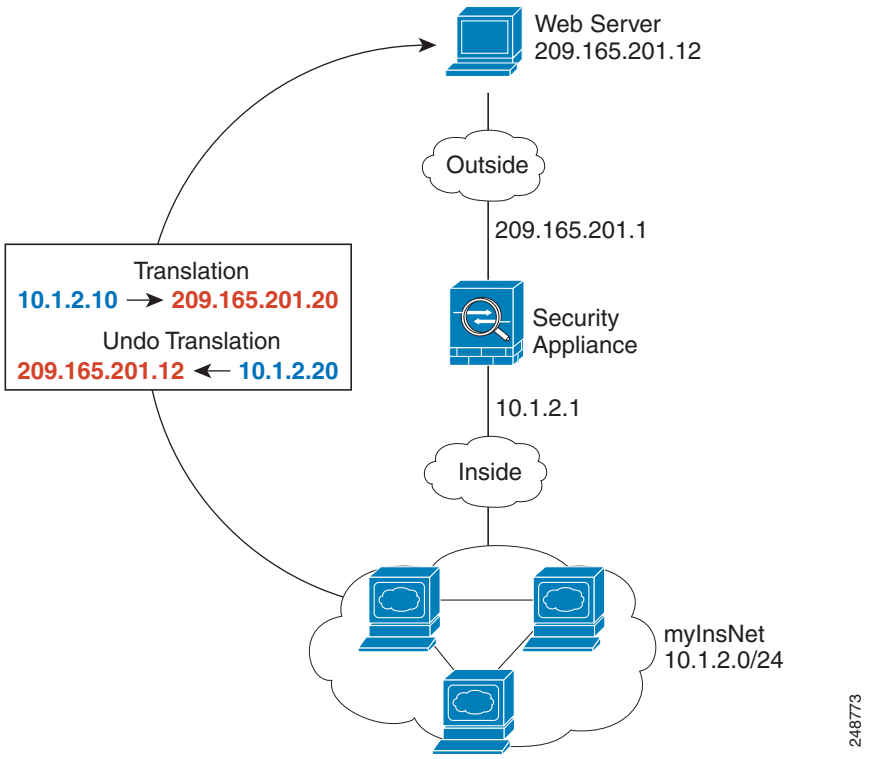
Step 2 Configure static NAT for the object:

```
hostname(config-network-object)# nat (inside,outside) static 209.165.201.10
```

NAT for Inside Hosts (Dynamic NAT) and NAT for an Outside Web Server (Static NAT)

The following example configures dynamic NAT for inside users on a private network when they access the outside. Also, when inside users connect to an outside web server, that web server address is translated to an address that appears to be on the inside network.

Figure 10-2 Dynamic NAT for Inside, Static NAT for Outside Web Server



Procedure

Step 1 Create a network object for the dynamic NAT pool to which you want to translate the inside addresses.

```
hostname(config)# object network myNatPool
hostname(config-network-object)# range 209.165.201.20 209.165.201.30
```

Step 2 Create a network object for the inside network.

```
hostname(config)# object network myInsNet
hostname(config-network-object)# subnet 10.1.2.0 255.255.255.0
```

Step 3 Enable dynamic NAT for the inside network using the dynamic NAT pool object.

```
hostname(config-network-object)# nat (inside,outside) dynamic myNatPool
```

Step 4 Create a network object for the outside web server.

```
hostname(config)# object network myWebServ
hostname(config-network-object)# host 209.165.201.12
```

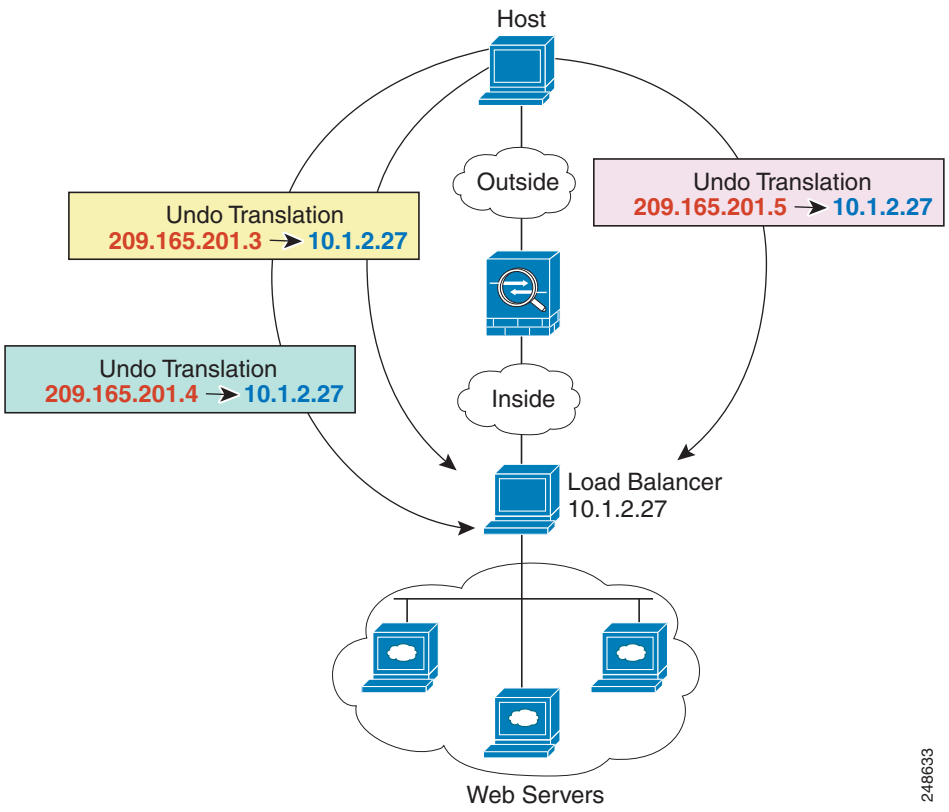
Step 5 Configure static NAT for the web server.

```
hostname(config-network-object)# nat (outside,inside) static 10.1.2.20
```

Inside Load Balancer with Multiple Mapped Addresses (Static NAT, One-to-Many)

The following example shows an inside load balancer that is translated to multiple IP addresses. When an outside host accesses one of the mapped IP addresses, it is untranslated to the single load balancer address. Depending on the URL requested, it redirects traffic to the correct web server.

Figure 10-3 Static NAT with One-to-Many for an Inside Load Balancer



Procedure

Step 1 Create a network object for the addresses to which you want to map the load balancer.

```
hostname(config)# object network myPublicIPs
hostname(config-network-object)# range 209.165.201.3 209.265.201.8
```

Step 2 Create a network object for the load balancer.

```
hostname(config)# object network myLBHost
hostname(config-network-object)# host 10.1.2.27
```

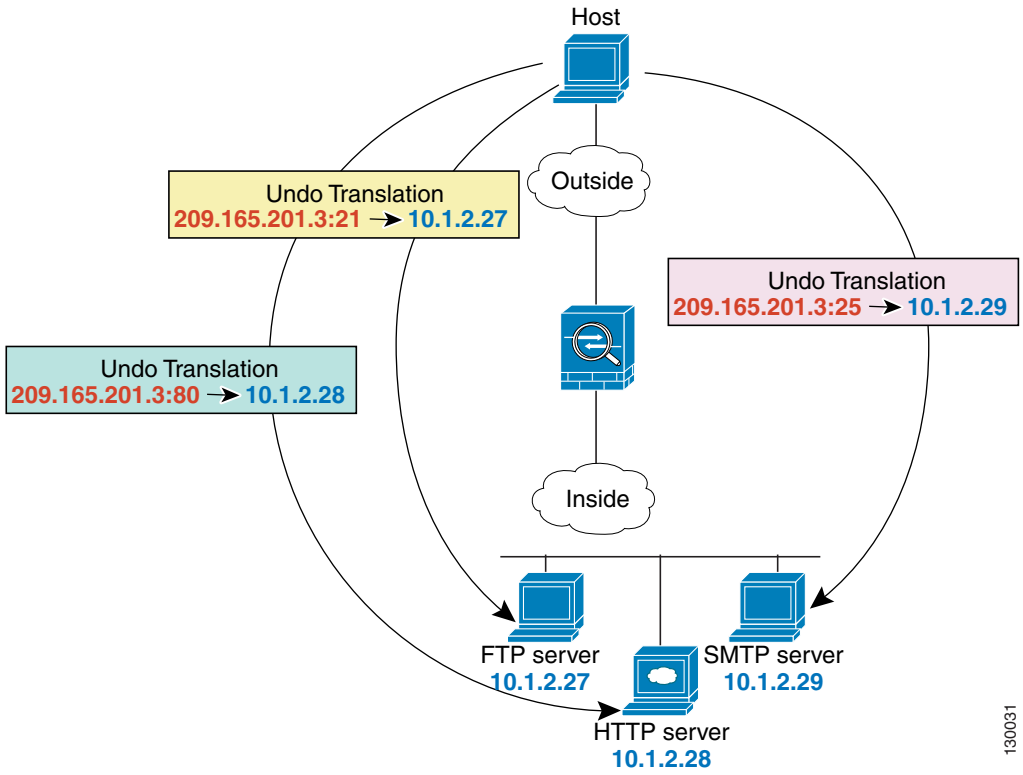
Step 3 Configure static NAT for the load balancer applying the range object.

```
hostname(config-network-object)# nat (inside,outside) static myPublicIPs
```

Single Address for FTP, HTTP, and SMTP (Static NAT-with-Port-Translation)

The following static NAT-with-port-translation example provides a single address for remote users to access FTP, HTTP, and SMTP. These servers are actually different devices on the real network, but for each server, you can specify static NAT-with-port-translation rules that use the same mapped IP address, but different ports.

Figure 10-4 Static NAT-with-Port-Translation



Procedure

Step 1 Create a network object for the FTP server and configure static NAT with port translation, mapping the FTP port to itself.

```
hostname(config)# object network FTP_SERVER
hostname(config-network-object)# host 10.1.2.27
hostname(config-network-object)# nat (inside,outside) static 209.165.201.3 service tcp ftp
ftp
```

Step 2 Create a network object for the HTTP server and configure static NAT with port translation, mapping the HTTP port to itself.

```
hostname(config)# object network HTTP_SERVER
hostname(config-network-object)# host 10.1.2.28
hostname(config-network-object)# nat (inside,outside) static 209.165.201.3 service tcp
http http
```

Step 3 Create a network object for the SMTP server and configure static NAT with port translation, mapping the SMTP port to itself.

```
hostname(config)# object network SMTP_SERVER
hostname(config-network-object)# host 10.1.2.29
hostname(config-network-object)# nat (inside,outside) static 209.165.201.3 service tcp
smtp smtp
```

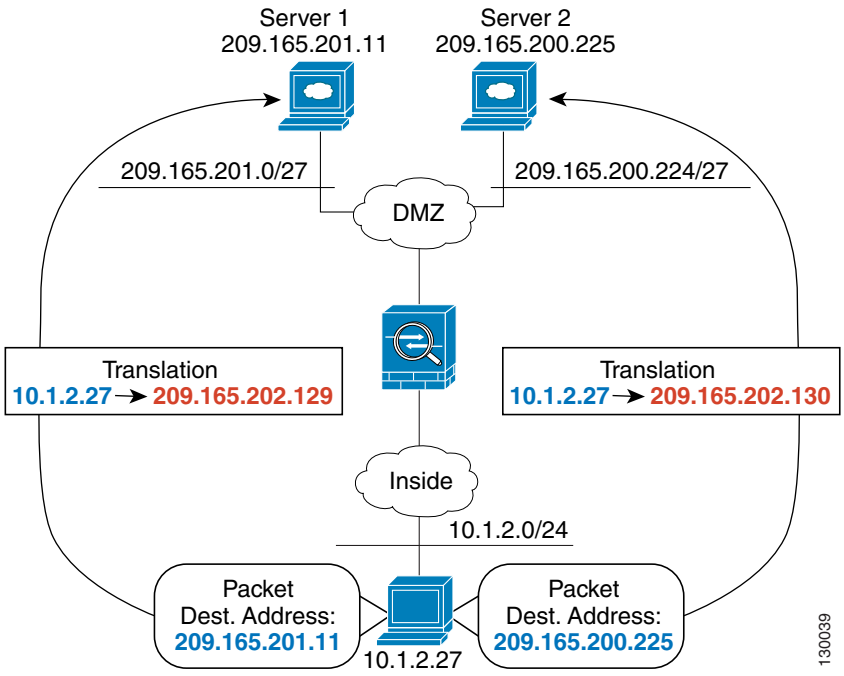
Examples for Twice NAT

This section includes the following configuration examples:

Different Translation Depending on the Destination (Dynamic Twice PAT)

The following figure shows a host on the 10.1.2.0/24 network accessing two different servers. When the host accesses the server at 209.165.201.11, the real address is translated to 209.165.202.129:*port*. When the host accesses the server at 209.165.200.225, the real address is translated to 209.165.202.130:*port*.

Figure 10-5 Twice NAT with Different Destination Addresses



Procedure

Step 1 Add a network object for the inside network:

```
hostname(config)# object network myInsideNetwork
hostname(config-network-object)# subnet 10.1.2.0 255.255.255.0
```

Step 2 Add a network object for the DMZ network 1:

```
hostname(config)# object network DMZnetwork1
hostname(config-network-object)# subnet 209.165.201.0 255.255.255.224
```

Step 3 Add a network object for the PAT address:

```
hostname(config)# object network PATaddress1
hostname(config-network-object)# host 209.165.202.129
```

Step 4 Configure the first twice NAT rule:

```
hostname(config)# nat (inside,dmz) source dynamic myInsideNetwork PATaddress1
destination static DMZnetwork1 DMZnetwork1
```

Because you do not want to translate the destination address, you need to configure identity NAT for it by specifying the same address for the real and mapped destination addresses.

Step 5 Add a network object for the DMZ network 2:

```
hostname(config)# object network DMZnetwork2
hostname(config-network-object)# subnet 209.165.200.224 255.255.255.224
```

Step 6 Add a network object for the PAT address:

```
hostname(config)# object network PATaddress2
hostname(config-network-object)# host 209.165.202.130
```

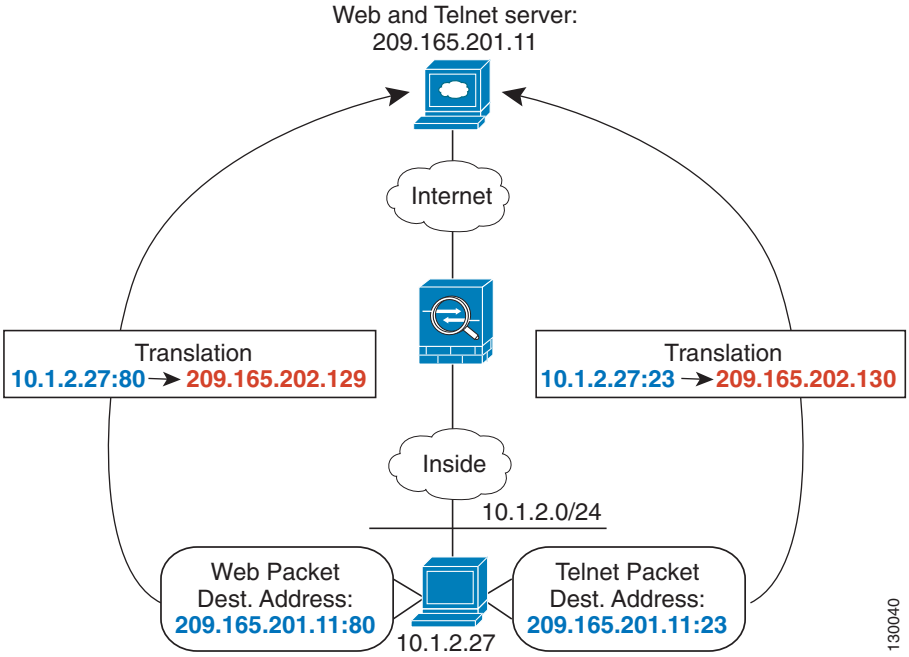
Step 7 Configure the second twice NAT rule:

```
hostname(config)# nat (inside,dmz) source dynamic myInsideNetwork PATaddress2
destination static DMZnetwork2 DMZnetwork2
```

Different Translation Depending on the Destination Address and Port (Dynamic PAT)

The following figure shows the use of source and destination ports. The host on the 10.1.2.0/24 network accesses a single host for both web services and Telnet services. When the host accesses the server for Telnet services, the real address is translated to 209.165.202.129:*port*. When the host accesses the same server for web services, the real address is translated to 209.165.202.130:*port*.

Figure 10-6 Twice NAT with Different Destination Ports



Procedure

Step 1 Add a network object for the inside network:

```
hostname(config)# object network myInsideNetwork
hostname(config-network-object)# subnet 10.1.2.0 255.255.255.0
```

Step 2 Add a network object for the Telnet/Web server:

```
hostname(config)# object network TelnetWebServer
hostname(config-network-object)# host 209.165.201.11
```

Step 3 Add a network object for the PAT address when using Telnet:

```
hostname(config)# object network PATaddress1
hostname(config-network-object)# host 209.165.202.129
```

Step 4 Add a service object for Telnet:

```
hostname(config)# object service TelnetObj
hostname(config-network-object)# service tcp destination eq telnet
```

Step 5 Configure the first twice NAT rule:

```
hostname(config)# nat (inside,outside) source dynamic myInsideNetwork PATaddress1
destination static TelnetWebServer TelnetWebServer service TelnetObj TelnetObj
```

Because you do not want to translate the destination address or port, you need to configure identity NAT for them by specifying the same address for the real and mapped destination addresses, and the same port for the real and mapped service.

Step 6 Add a network object for the PAT address when using HTTP:

```
hostname(config)# object network PATaddress2
hostname(config-network-object)# host 209.165.202.130
```

Step 7 Add a service object for HTTP:

```
hostname(config)# object service HTTPObj
hostname(config-network-object)# service tcp destination eq http
```

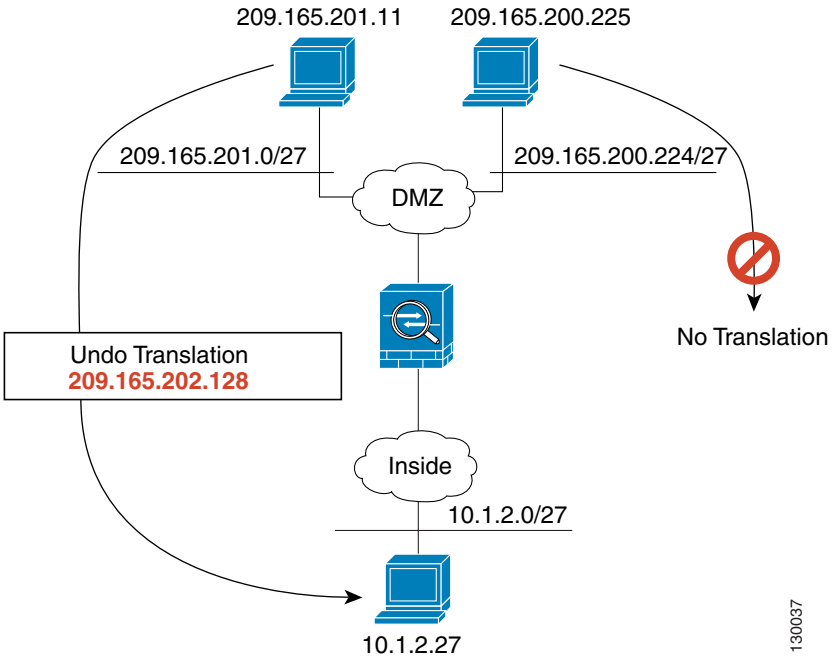
Step 8 Configure the second twice NAT rule:

```
hostname(config)# nat (inside,outside) source dynamic myInsideNetwork PATaddress2
destination static TelnetWebServer TelnetWebServer service HTTPObj HTTPObj
```

Example: Twice NAT with Destination Address Translation

The following figure shows a remote host connecting to a mapped host. The mapped host has a twice static NAT translation that translates the real address only for traffic to and from the 209.165.201.0/27 network. A translation does not exist for the 209.165.200.224/27 network, so the translated host cannot connect to that network, nor can a host on that network connect to the translated host.

Figure 10-7 Twice Static NAT with Destination Address Translation



NAT in Routed and Transparent Mode

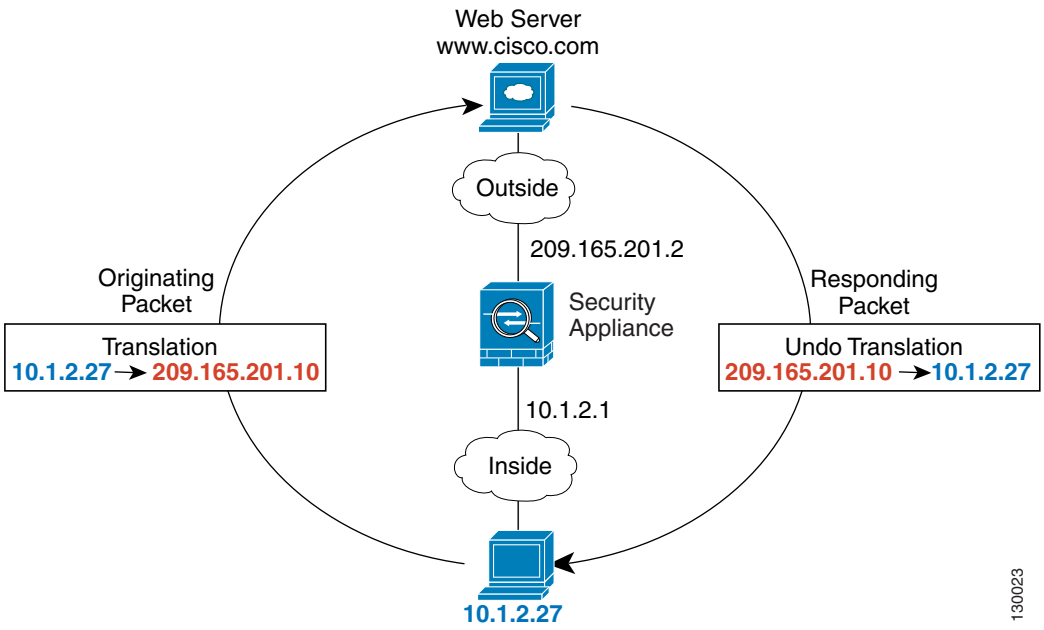
You can configure NAT in both routed and transparent firewall mode. This section describes typical usage for each firewall mode.

- NAT in Routed Mode, page 10-10
- NAT in Transparent Mode, page 10-10

NAT in Routed Mode

The following figure shows a typical NAT example in routed mode, with a private network on the inside.

Figure 10-8 NAT Example: Routed Mode



1. When the inside host at 10.1.2.27 sends a packet to a web server, the real source address of the packet, 10.1.2.27, is changed to a mapped address, 209.165.201.10.
2. When the server responds, it sends the response to the mapped address, 209.165.201.10, and the ASA receives the packet because the ASA performs proxy ARP to claim the packet.
3. The ASA then changes the translation of the mapped address, 209.165.201.10, back to the real address, 10.1.2.27, before sending it to the host.

NAT in Transparent Mode

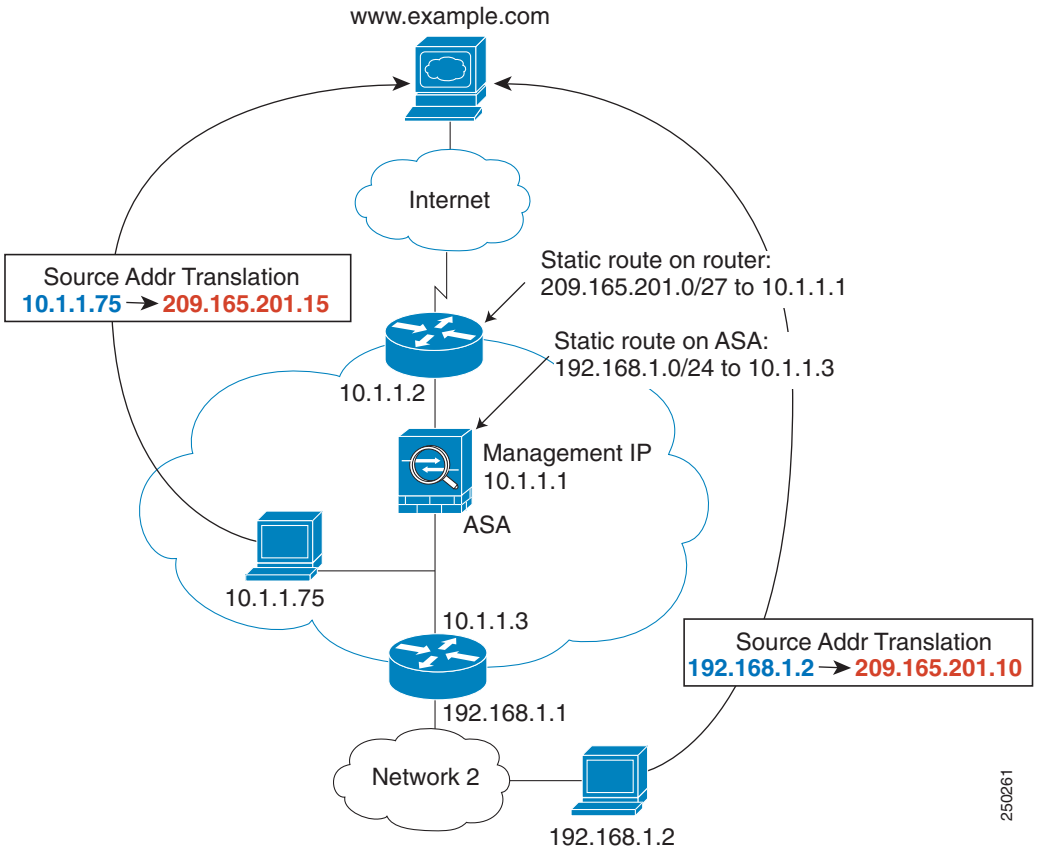
Using NAT in transparent mode eliminates the need for the upstream or downstream routers to perform NAT for their networks.

NAT in transparent mode has the following requirements and limitations:

- Because the transparent firewall does not have any interface IP addresses, you cannot use interface PAT.
- ARP inspection is not supported. Moreover, if for some reason a host on one side of the ASA sends an ARP request to a host on the other side of the ASA, and the initiating host real address is mapped to a different address on the same subnet, then the real address remains visible in the ARP request.
- Translating between IPv4 and IPv6 networks is not supported. Translating between two IPv6 networks, or between two IPv4 networks is supported.

The following figure shows a typical NAT scenario in transparent mode, with the same network on the inside and outside interfaces. The transparent firewall in this scenario is performing the NAT service so that the upstream router does not have to perform NAT.

Figure 10-9 NAT Example: Transparent Mode



1. When the inside host at 10.1.1.75 sends a packet to a web server, the real source address of the packet, 10.1.1.75, is changed to a mapped address, 209.165.201.15.
2. When the server responds, it sends the response to the mapped address, 209.165.201.15, and the ASA receives the packet because the upstream router includes this mapped network in a static route directed to the ASA management IP address. See Mapped Addresses and Routing, page 10-12 for more information about required routes.
3. The ASA then undoes the translation of the mapped address, 209.165.201.15, back to the real address, 10.1.1.1.75. Because the real address is directly-connected, the ASA sends it directly to the host.
4. For host 192.168.1.2, the same process occurs, except for returning traffic, the ASA looks up the route in its routing table and sends the packet to the downstream router at 10.1.1.3 based on the ASA static route for 192.168.1.0/24. See Transparent Mode Routing Requirements for Remote Networks, page 10-14 for more information about required routes.

Routing NAT Packets

The ASA needs to be the destination for any packets sent to the mapped address. The ASA also needs to determine the egress interface for any packets it receives destined for mapped addresses. This section describes how the ASA handles accepting and delivering packets with NAT.

- Mapped Addresses and Routing, page 10-12

- Transparent Mode Routing Requirements for Remote Networks, page 10-14
- Determining the Egress Interface, page 10-14

Mapped Addresses and Routing

When you translate the real address to a mapped address, the mapped address you choose determines how to configure routing, if necessary, for the mapped address.

See additional guidelines about mapped IP addresses in Additional Guidelines for NAT, page 9-8.

The following topics explain the mapped address types:

- Addresses on the Same Network as the Mapped Interface, page 10-12
- Addresses on a Unique Network, page 10-12
- The Same Address as the Real Address (Identity NAT), page 10-13

Addresses on the Same Network as the Mapped Interface

If you use addresses on the same network as the mapped interface, the ASA uses proxy ARP to answer any ARP requests for the mapped addresses, thus intercepting traffic destined for a mapped address. This solution simplifies routing because the ASA does not have to be the gateway for any additional networks. This solution is ideal if the outside network contains an adequate number of free addresses, a consideration if you are using a 1:1 translation like dynamic NAT or static NAT. Dynamic PAT greatly extends the number of translations you can use with a small number of addresses, so even if the available addresses on the outside network is small, this method can be used. For PAT, you can even use the IP address of the mapped interface.

Note If you configure the mapped interface to be any interface, and you specify a mapped address on the same network as one of the mapped interfaces, then if an ARP request for that mapped address comes in on a *different* interface, then you need to manually configure an ARP entry for that network on the ingress interface, specifying its MAC address (see the **arp** command). Typically, if you specify any interface for the mapped interface, then you use a unique network for the mapped addresses, so this situation would not occur.

Addresses on a Unique Network

If you need more addresses than are available on the mapped interface network, you can identify addresses on a different subnet. The upstream router needs a static route for the mapped addresses that points to the ASA. Alternatively for routed mode, you can configure a static route on the ASA for the mapped addresses using any IP address on the destination network as the gateway, and then redistribute the route using your routing protocol. For example, if you use NAT for the inside network (10.1.1.0/24) and use the mapped IP address 209.165.201.5, then you can configure the following static route that can be redistributed:

```
route inside 209.165.201.5 255.255.255.255 10.1.1.99
```

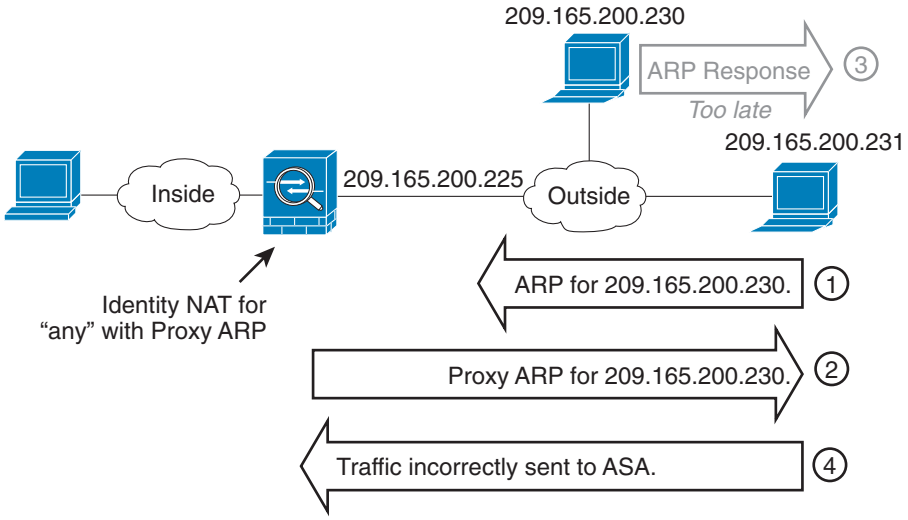
For transparent mode, if the real host is directly-connected, configure the static route on the upstream router to point to the ASA: specify the bridge group IP address. For remote hosts in transparent mode, in the static route on the upstream router, you can alternatively specify the downstream router IP address.

The Same Address as the Real Address (Identity NAT)

The default behavior for identity NAT has proxy ARP enabled, matching other static NAT rules. You can disable proxy ARP if desired. You can also disable proxy ARP for regular static NAT if desired, in which case you need to be sure to have proper routes on the upstream router.

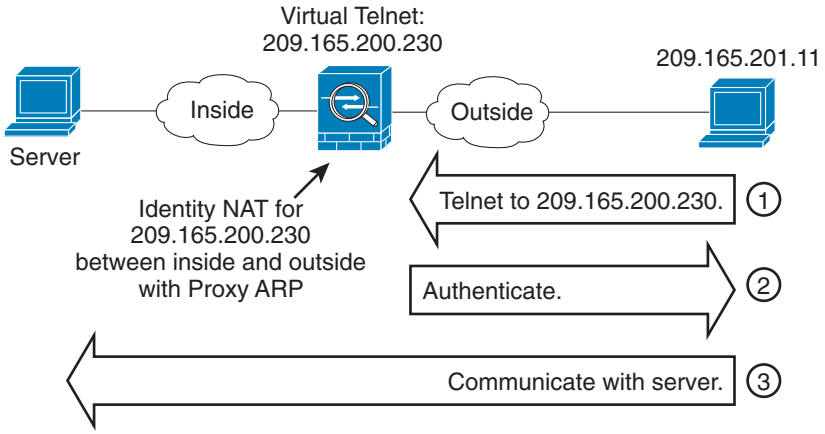
Normally for identity NAT, proxy ARP is not required, and in some cases can cause connectivity issues. For example, if you configure a broad identity NAT rule for "any" IP address, then leaving proxy ARP enabled can cause problems for hosts on the network directly connected to the mapped interface. In this case, when a host on the mapped network wants to communicate with another host on the same network, then the address in the ARP request matches the NAT rule (which matches "any" address). The ASA will then proxy ARP for the address, even though the packet is not actually destined for the ASA. (Note that this problem occurs even if you have a twice NAT rule; although the NAT rule must match both the source and destination addresses, the proxy ARP decision is made only on the "source" address). If the ASA ARP response is received before the actual host ARP response, then traffic will be mistakenly sent to the ASA (see the following figure).

Figure 10-10 Proxy ARP Problems with Identity NAT



In rare cases, you need proxy ARP for identity NAT; for example for virtual Telnet. When using AAA for network access, a host needs to authenticate with the ASA using a service like Telnet before any other traffic can pass. You can configure a virtual Telnet server on the ASA to provide the necessary login. When accessing the virtual Telnet address from the outside, you must configure an identity NAT rule for the address specifically for the proxy ARP functionality. Due to internal processes for virtual Telnet, proxy ARP lets the ASA keep traffic destined for the virtual Telnet address rather than send the traffic out the source interface according to the NAT rule. (See the following figure).

Figure 10-11 Proxy ARP and Virtual Telnet



Transparent Mode Routing Requirements for Remote Networks

When you use NAT in transparent mode, some types of traffic require static routes. See the general operations configuration guide for more information.

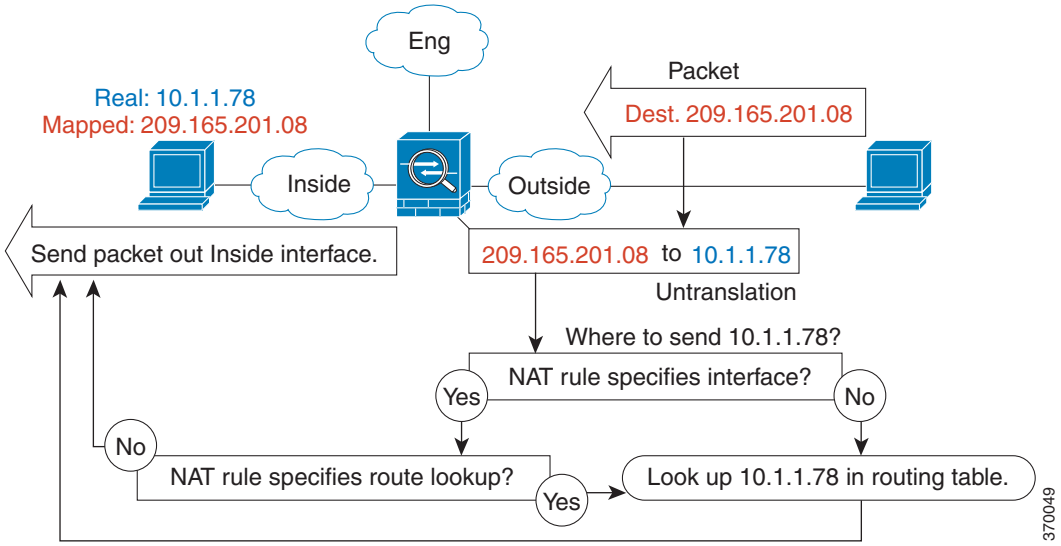
Determining the Egress Interface

When the ASA receives traffic for a mapped address, the ASA untranslates the destination address according to the NAT rule, and then it sends the packet on to the real address. The ASA determines the egress interface for the packet in the following ways:

- Transparent mode—The ASA determines the egress interface for the real address by using the NAT rule; you must specify the source and destination interfaces as part of the NAT rule.
- Routed mode—The ASA determines the egress interface in one of the following ways:
  - You configure the interface in the NAT rule—The ASA uses the NAT rule to determine the egress interface. However, you have the option to always use a route lookup instead. In certain scenarios, a route lookup override is required; for example, see NAT and VPN Management Access, page 10-19.
  - You do not configure the interface in the NAT rule—The ASA uses a route lookup to determine the egress interface.

The following figure shows the egress interface selection method in routed mode. In almost all cases, a route lookup is equivalent to the NAT rule interface, but in some configurations, the two methods might differ.

Figure 10-12 Routed Mode Egress Interface Selection



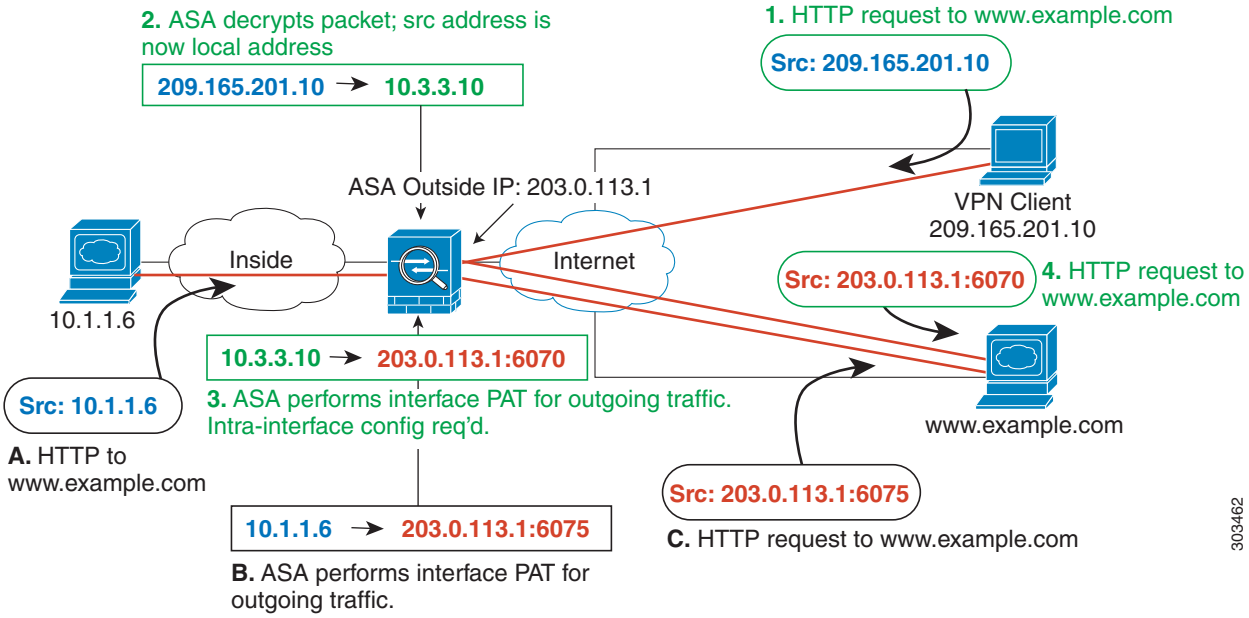
NAT for VPN

The following topics explain NAT usage with the various types of VPN.

NAT and Remote Access VPN

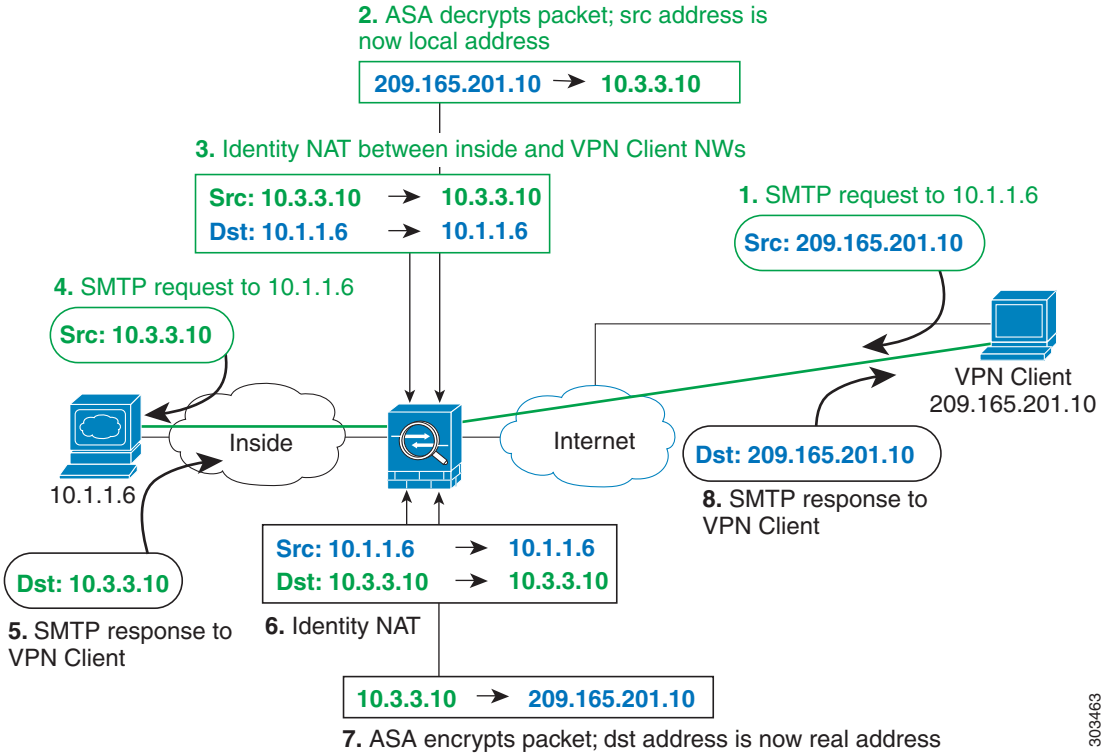
The following figure shows both an inside server (10.1.1.6) and a VPN client (209.165.201.10) accessing the Internet. Unless you configure split tunneling for the VPN client (where only specified traffic goes through the VPN tunnel), then Internet-bound VPN traffic must also go through the ASA. When the VPN traffic enters the ASA, the ASA decrypts the packet; the resulting packet includes the VPN client local address (10.3.3.10) as the source. For both inside and VPN client local networks, you need a public IP address provided by NAT to access the Internet. The below example uses interface PAT rules. To allow the VPN traffic to exit the same interface it entered, you also need to enable intra-interface communication (also known as “hairpin” networking).

Figure 10-13 Interface PAT for Internet-Bound VPN Traffic (Intra-Interface)



The following figure shows a VPN client that wants to access an inside mail server. Because the ASA expects traffic between the inside network and any outside network to match the interface PAT rule you set up for Internet access, traffic from the VPN client (10.3.3.10) to the SMTP server (10.1.1.6) will be dropped due to a reverse path failure: traffic from 10.3.3.10 to 10.1.1.6 does not match a NAT rule, but returning traffic from 10.1.1.6 to 10.3.3.10 *should* match the interface PAT rule for outgoing traffic. Because forward and reverse flows do not match, the ASA drops the packet when it is received. To avoid this failure, you need to exempt the inside-to-VPN client traffic from the interface PAT rule by using an identity NAT rule between those networks. Identity NAT simply translates an address to the same address.

Figure 10-14 Identity NAT for VPN Clients



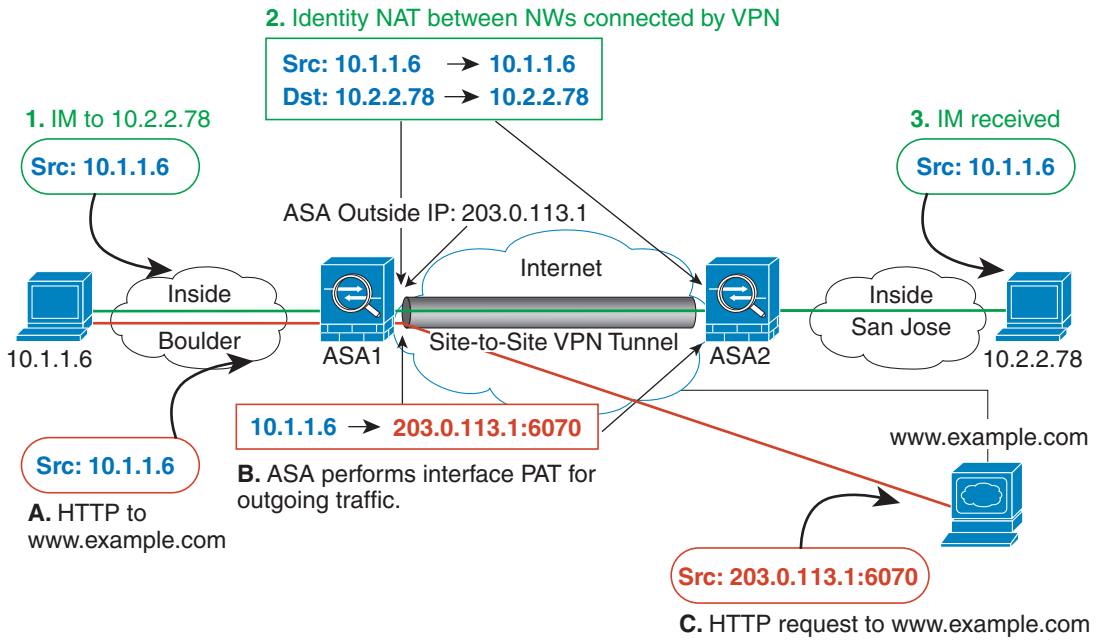
See the following sample NAT configuration for the above network:

```
! Enable hairpin for non-split-tunneled VPN client traffic:
same-security-traffic permit intra-interface

! Identify local VPN network, & perform object interface PAT when going to Internet:
object network vpn_local
    subnet 10.3.3.0 255.255.255.0
    nat (outside,outside) dynamic interface

! Identify inside network, & perform object interface PAT when going to Internet:
object network inside_nw
    subnet 10.1.1.0 255.255.255.0
    nat (inside,outside) dynamic interface

! Use twice NAT to pass traffic between the inside network and the VPN client without
! address translation (identity NAT):
nat (inside,outside) source static inside_nw inside_nw destination static vpn_local
vpn_local
```

NAT and Site-to-Site VPN

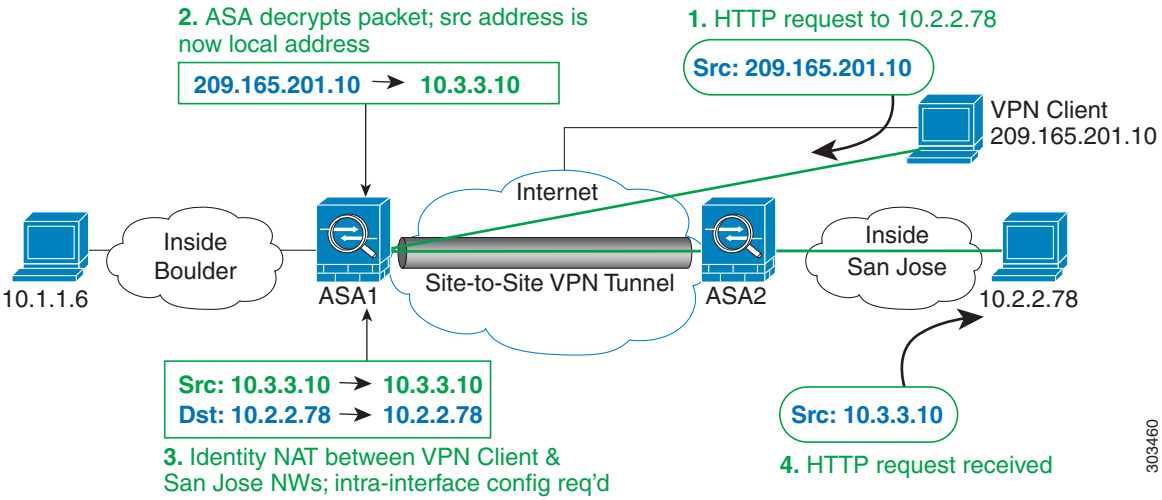
The following figure shows a site-to-site tunnel connecting the Boulder and San Jose offices. For traffic that you want to go to the Internet (for example from 10.1.1.6 in Boulder to www.example.com), you need a public IP address provided by NAT to access the Internet. The below example uses interface PAT rules. However, for traffic that you want to go over the VPN tunnel (for example from 10.1.1.6 in Boulder to 10.2.2.78 in San Jose), you do not want to perform NAT; you need to exempt that traffic by creating an identity NAT rule. Identity NAT simply translates an address to the same address.

Figure 10-15 Interface PAT and Identity NAT for Site-to-Site VPN



The following figure shows a VPN client connected to ASA1 (Boulder), with a Telnet request for a server (10.2.2.78) accessible over a site-to-site tunnel between ASA1 and ASA2 (San Jose). Because this is a hairpin connection, you need to enable intra-interface communication, which is also required for non-split-tunneled Internet-bound traffic from the VPN client. You also need to configure identity NAT between the VPN client and the Boulder & San Jose networks, just as you would between any networks connected by VPN to exempt this traffic from outbound NAT rules.

Figure 10-16 VPN Client Access to Site-to-Site VPN



See the following sample NAT configuration for ASA1 (Boulder):

```
! Enable hairpin for VPN client traffic:
same-security-traffic permit intra-interface

! Identify local VPN network, & perform object interface PAT when going to Internet:
```

```
object network vpn_local
    subnet 10.3.3.0 255.255.255.0
    nat (outside,outside) dynamic interface

! Identify inside Boulder network, & perform object interface PAT when going to Internet:
object network boulder_inside
    subnet 10.1.1.0 255.255.255.0
    nat (inside,outside) dynamic interface

! Identify inside San Jose network for use in twice NAT rule:
object network sanjose_inside
    subnet 10.2.2.0 255.255.255.0

! Use twice NAT to pass traffic between the Boulder network and the VPN client without
! address translation (identity NAT):
nat (inside,outside) source static boulder_inside boulder_inside destination static
vpn_local vpn_local

! Use twice NAT to pass traffic between the Boulder network and San Jose without
! address translation (identity NAT):
nat (inside,outside) source static boulder_inside boulder_inside destination static
sanjose_inside sanjose_inside

! Use twice NAT to pass traffic between the VPN client and San Jose without
! address translation (identity NAT):
nat (outside,outside) source static vpn_local vpn_local destination static sanjose_inside
sanjose_inside
```

See the following sample NAT configuration for ASA2 (San Jose):

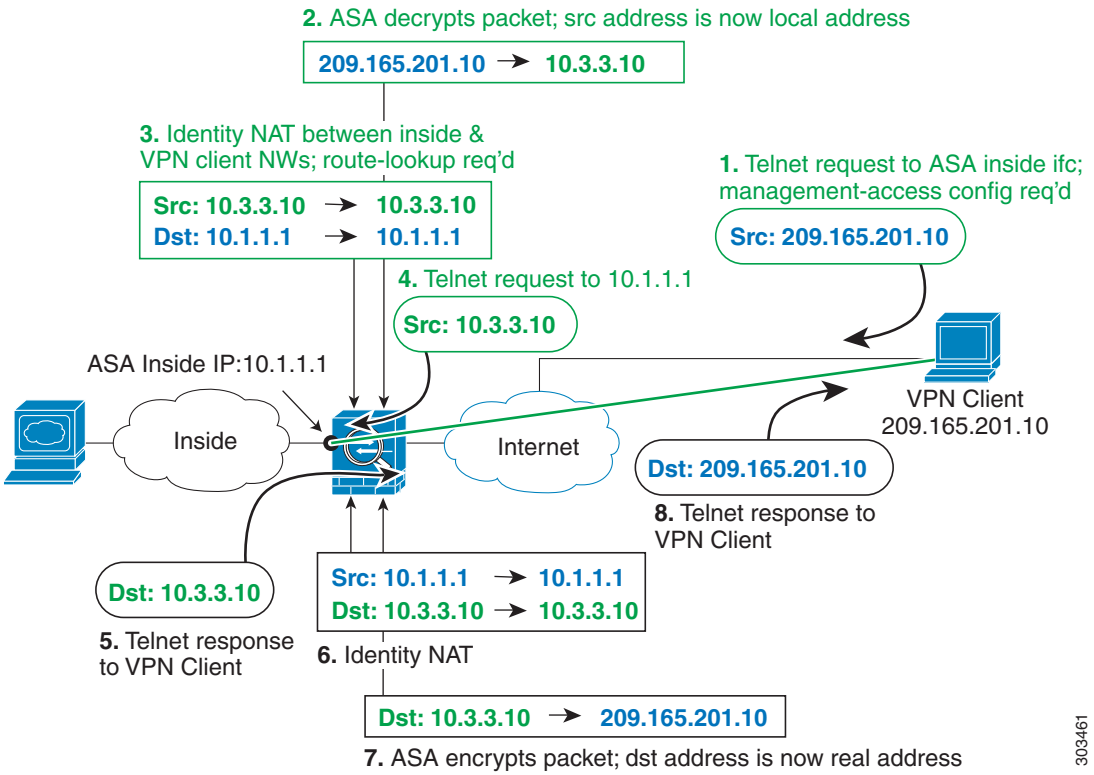
```
! Identify inside San Jose network, & perform object interface PAT when going to Internet:
object network sanjose_inside
    subnet 10.2.2.0 255.255.255.0
    nat (inside,outside) dynamic interface

! Identify inside Boulder network for use in twice NAT rule:
object network boulder_inside
    subnet 10.1.1.0 255.255.255.0

! Identify local VPN network for use in twice NAT rule:
object network vpn_local
    subnet 10.3.3.0 255.255.255.0

! Use twice NAT to pass traffic between the San Jose network and Boulder without
! address translation (identity NAT):
nat (inside,outside) source static sanjose_inside sanjose_inside destination static
boulder_inside boulder_inside

! Use twice NAT to pass traffic between the San Jose network and the VPN client without
! address translation (identity NAT):
nat (inside,outside) source static sanjose_inside sanjose_inside destination static
vpn_local vpn_local
```

NAT and VPN Management Access

When using VPN, you can allow management access to an interface other than the one from which you entered the ASA (see the **management-access** command). For example, if you enter the ASA from the outside interface, the management-access feature lets you connect to the inside interface using ASDM, SSH, Telnet, or SNMP; or you can ping the inside interface.

The following figure shows a VPN client Telnetting to the ASA inside interface. When you use a management-access interface, and you configure identity NAT according to NAT and Remote Access VPN, page 10-15 or NAT and Site-to-Site VPN, page 10-17, you must configure NAT with the route lookup option. Without route lookup, the ASA sends traffic out the interface specified in the NAT command, regardless of what the routing table says; in the below example, the egress interface is the inside interface. You do not want the ASA to send the management traffic out to the inside network; it will never return to the inside interface IP address. The route lookup option lets the ASA send the traffic directly to the inside interface IP address instead of to the inside network. For traffic from the VPN client to a host on the inside network, the route lookup option will still result in the correct egress interface (inside), so normal traffic flow is not affected. See the Determining the Egress Interface, page 10-14 for more information about the route lookup option.

Figure 10-17 VPN Management Access



See the following sample NAT configuration for the above network:

```
! Enable hairpin for non-split-tunneled VPN client traffic:
same-security-traffic permit intra-interface

! Enable management access on inside ifc:
management-access inside

! Identify local VPN network, & perform object interface PAT when going to Internet:
object network vpn_local
    subnet 10.3.3.0 255.255.255.0
    nat (outside,outside) dynamic interface

! Identify inside network, & perform object interface PAT when going to Internet:
object network inside_nw
    subnet 10.1.1.0 255.255.255.0
    nat (inside,outside) dynamic interface
```

```
! Use twice NAT to pass traffic between the inside network and the VPN client without
! address translation (identity NAT), w/route-lookup:
nat (outside,inside) source static vpn_local vpn_local destination static inside_nw
inside_nw route-lookup
```

Troubleshooting NAT and VPN

See the following monitoring tools for troubleshooting NAT issues with VPN:

- Packet tracer—When used correctly, a packet tracer shows which NAT rules a packet is hitting.
- **show nat detail**—Shows hit counts and untranslated traffic for a given NAT rule.
- **show conn all**—Lets you see active connections including to and from the box traffic.

To familiarize yourself with a non-working configuration vs. a working configuration, you can perform the following steps:

1. Configure VPN without identity NAT.
2. Enter **show nat detail** and **show conn all**.
3. Add the identity NAT configuration.
4. Repeat **show nat detail** and **show conn all**.

DNS and NAT

You might need to configure the ASA to modify DNS replies by replacing the address in the reply with an address that matches the NAT configuration. You can configure DNS modification when you configure each translation rule.

This feature rewrites the address in DNS queries and replies that match a NAT rule (for example, the A record for IPv4, the AAAA record for IPv6, or the PTR record for reverse DNS queries). For DNS replies traversing from a mapped interface to any other interface, the record is rewritten from the mapped value to the real value. Inversely, for DNS replies traversing from any interface to a mapped interface, the record is rewritten from the real value to the mapped value.

Following are some limitations with DNS rewrite:

- DNS rewrite is not applicable for PAT because multiple PAT rules are applicable for each A-record, and the PAT rule to use is ambiguous.
- If you configure a twice NAT rule, you cannot configure DNS modification if you specify the source address as well as the destination address. These kinds of rules can potentially have a different translation for a single address when going to A vs. B. Therefore, the ASA cannot accurately match the IP address inside the DNS reply to the correct twice NAT rule; the DNS reply does not contain information about which source/destination address combination was in the packet that prompted the DNS request.
- DNS rewrite requires DNS application inspection to be enabled, which it is on by default. See DNS Inspection, page 13-1 for more information.
- DNS rewrite is actually done on the xlate entry, not the NAT rule. Thus, if there is no xlate for a dynamic rule, rewrite cannot be done correctly. The same problem does not occur for static NAT.

The following topics provide examples of DNS rewrite:

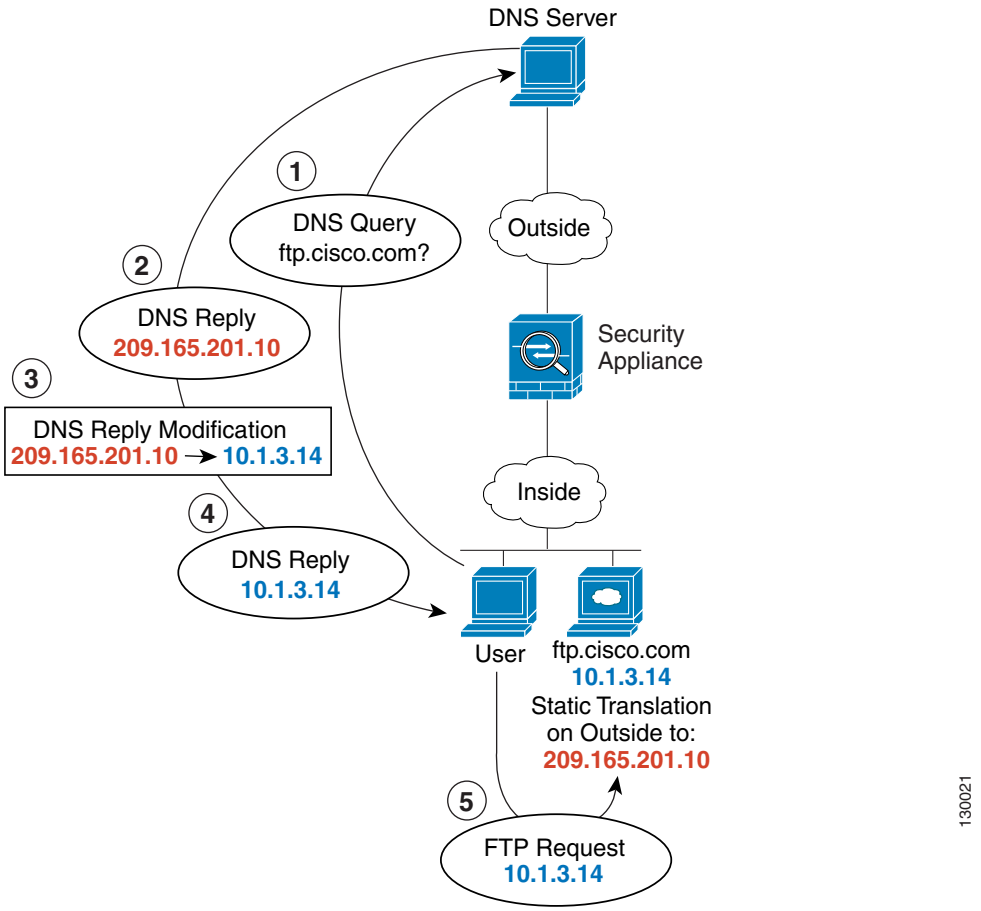
DNS Reply Modification, DNS Server on Outside

The following figure shows a DNS server that is accessible from the outside interface. A server, ftp.cisco.com, is on the inside interface. You configure the ASA to statically translate the ftp.cisco.com real address (10.1.3.14) to a mapped address (209.165.201.10) that is visible on the outside network.

In this case, you want to enable DNS reply modification on this static rule so that inside users who have access to ftp.cisco.com using the real address receive the real address from the DNS server, and not the mapped address.

When an inside host sends a DNS request for the address of ftp.cisco.com, the DNS server replies with the mapped address (209.165.201.10). The ASA refers to the static rule for the inside server and translates the address inside the DNS reply to 10.1.3.14. If you do not enable DNS reply modification, then the inside host attempts to send traffic to 209.165.201.10 instead of accessing ftp.cisco.com directly.

Figure 10-18 DNS Reply Modification, DNS Server on Outside



Procedure

Step 1 Create a network object for the FTP server.

```
hostname(config)# object network FTP_SERVER
hostname(config-network-object)# host 10.1.3.14
```

Step 2 Configure static NAT with DNS modification.

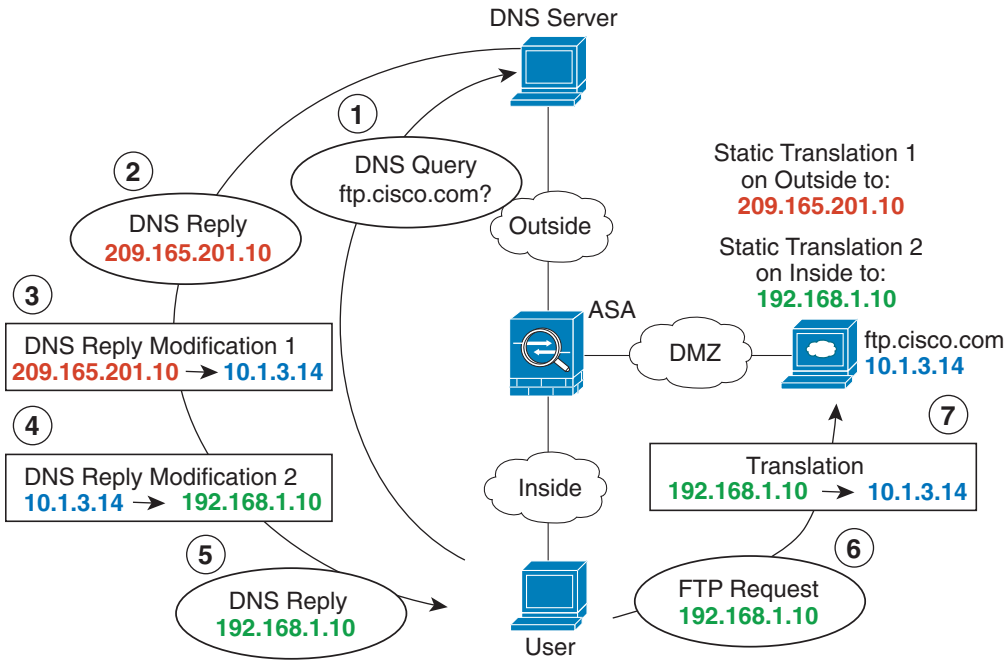
```
hostname(config-network-object)# nat (inside,outside) static 209.165.201.10 dns
```

DNS Reply Modification, DNS Server, Host, and Server on Separate Networks

The following figure shows a user on the inside network requesting the IP address for ftp.cisco.com, which is on the DMZ network, from an outside DNS server. The DNS server replies with the mapped address (209.165.201.10) according to the static rule between outside and DMZ even though the user is not on the DMZ network. The ASA translates the address inside the DNS reply to 10.1.3.14.

If the user needs to access ftp.cisco.com using the real address, then no further configuration is required. If there is also a static rule between the inside and DMZ, then you also need to enable DNS reply modification on this rule. The DNS reply will then be modified two times.In this case, the ASA again translates the address inside the DNS reply to 192.168.1.10 according to the static rule between inside and DMZ.

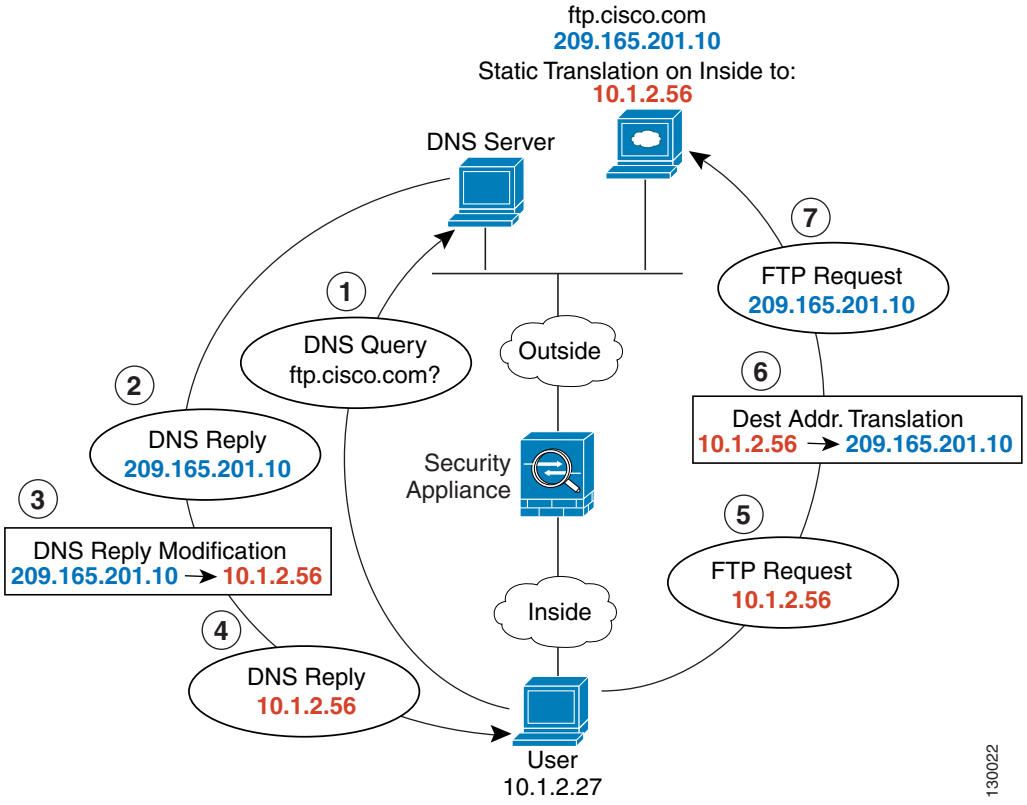
Figure 10-19 DNS Reply Modification, DNS Server, Host, and Server on Separate Networks



DNS Reply Modification, DNS Server on Host Network

The following figure shows an FTP server and DNS server on the outside. The ASA has a static translation for the outside server. In this case, when an inside user requests the address for ftp.cisco.com from the DNS server, the DNS server responds with the real address, 209.165.20.10. Because you want inside users to use the mapped address for ftp.cisco.com (10.1.2.56) you need to configure DNS reply modification for the static translation.

Figure 10-20 DNS Reply Modification, DNS Server on Host Network



Procedure

Step 1 Create a network object for the FTP server.

```
hostname(config)# object network FTP_SERVER
hostname(config-network-object)# host 209.165.201.10
```

Step 2 Configure static NAT with DNS modification.

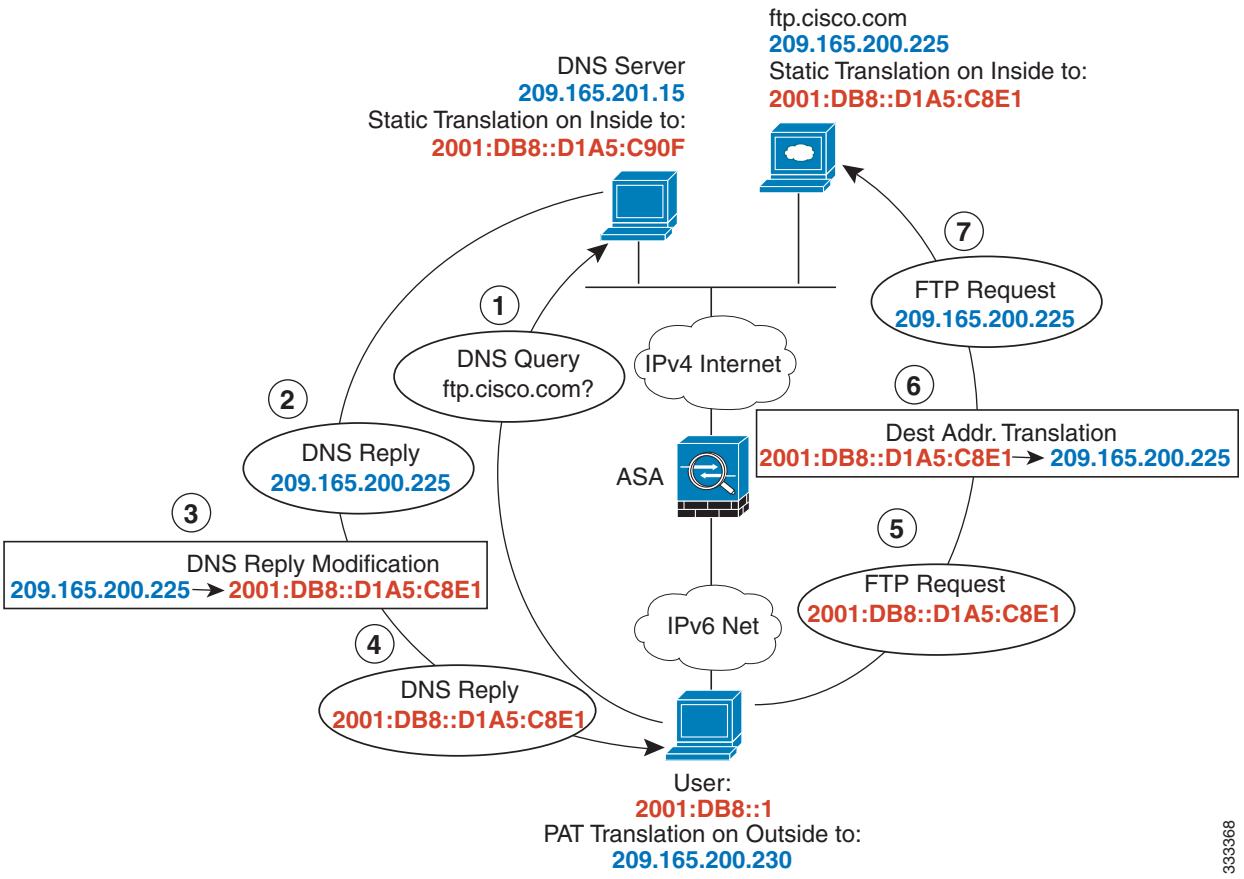
```
hostname(config-network-object)# nat (outside,inside) static 10.1.2.56 dns
```

DNS64 Reply Modification Using Outside NAT

The following figure shows an FTP server and DNS server on the outside IPv4 network. The ASA has a static translation for the outside server. In this case, when an inside IPv6 user requests the address for ftp.cisco.com from the DNS server, the DNS server responds with the real address, 209.165.200.225.

Because you want inside users to use the mapped address for ftp.cisco.com (2001:DB8::D1A5:C8E1) you need to configure DNS reply modification for the static translation. This example also includes a static NAT translation for the DNS server, and a PAT rule for the inside IPv6 hosts.

Figure 10-21 DNS64 Reply Modification Using Outside NAT



Procedure

Step 1 Create a network object for the FTP server and configure static NAT with DNS modification. Because this is a one-to-one translation, include the **net-to-net** option for NAT46.

```
hostname(config)# object network FTP_SERVER
hostname(config-network-object)# host 209.165.200.225
hostname(config-network-object)# nat (outside,inside) static 2001:DB8::D1A5:C8E1/128
net-to-net dns
```

Step 2 Create a network object for the DNS server and configure static NAT. Include the **net-to-net** option for NAT46.

```
hostname(config)# object network DNS_SERVER
hostname(config-network-object)# host 209.165.201.15
hostname(config-network-object)# nat (outside,inside) static 2001:DB8::D1A5:C90F/128
net-to-net
```

Step 3 Configure an IPv4 PAT pool for translating the inside IPv6 network.

```
hostname(config)# object network IPv4_POOL
hostname(config-network-object)# range 203.0.113.1 203.0.113.254
```

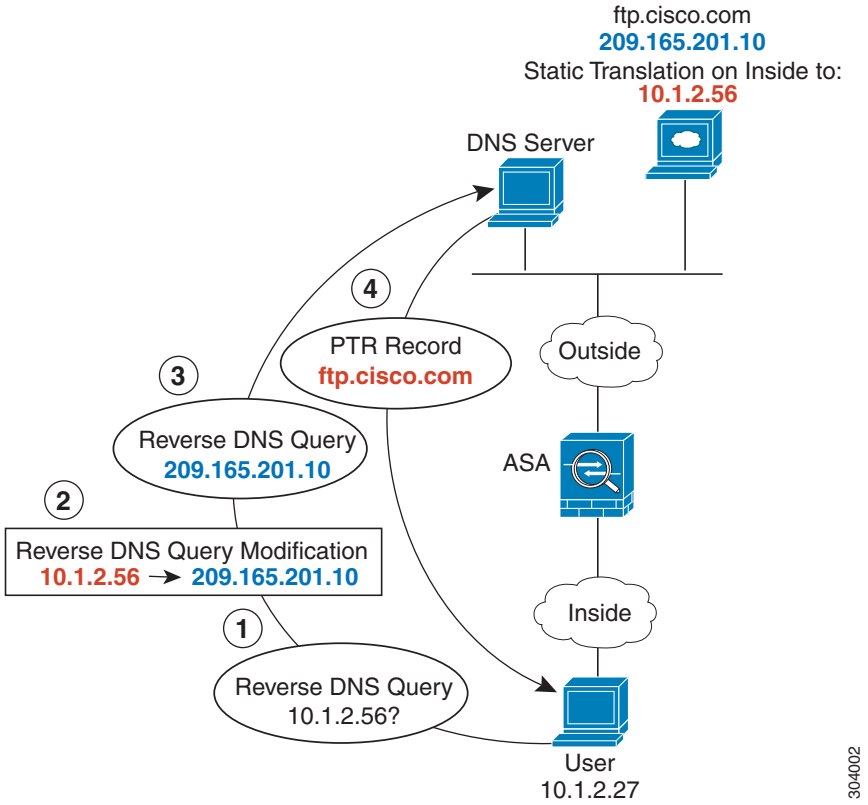
Step 4 Create a network object for the inside IPv6 network, and configure dynamic NAT with a PAT pool.

```
hostname(config)# object network IPv6_INSIDE
hostname(config-network-object)# subnet 2001:DB8::/96
hostname(config-network-object)# nat (inside,outside) dynamic pat-pool IPv4_POOL
```

PTR Modification, DNS Server on Host Network

The following figure shows an FTP server and DNS server on the outside. The ASA has a static translation for the outside server. In this case, when an inside user performs a reverse DNS lookup for 10.1.2.56, the ASA modifies the reverse DNS query with the real address, and the DNS server responds with the server name, ftp.cisco.com.

Figure 10-22 PTR Modification, DNS Server on Host Network



PART 3

Service Policies and Application Inspection

Service Policy Using the Modular Policy Framework

Service policies using Modular Policy Framework provide a consistent and flexible way to configure ASA features. For example, you can use a service policy to create a timeout configuration that is specific to a particular TCP application, as opposed to one that applies to all TCP applications. A service policy consists of multiple actions or rules applied to an interface or applied globally.

About Service Policies

The following topics describe how service policies work.

The Components of a Service Policy

The point of service policies is to apply advanced services to the traffic you are allowing. Any traffic permitted by access rules can have service policies applied, and thus receive special processing, such as being redirected to a service module or having application inspection applied.

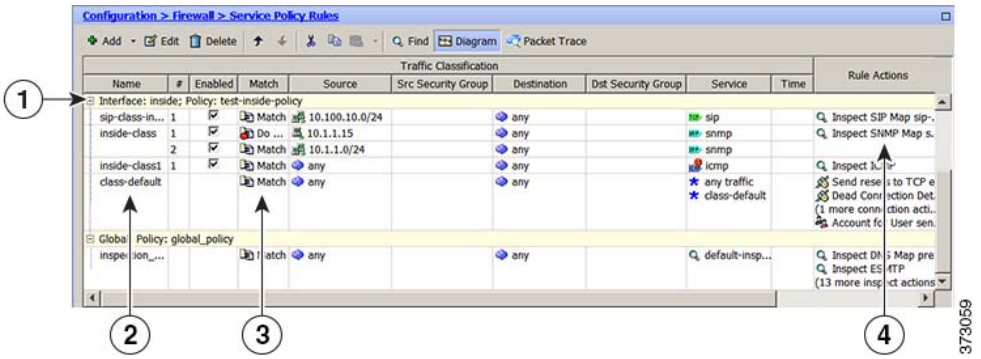
You can have these types of service policy:

- One global policy that gets applied to all interfaces.
- One service policy applied per interface. The policy can be a mix of classes for traffic going through the device and management traffic directed at the ASA interface rather than going through it,

Each service policy is composed of the following elements:

1. Service policy map, which is the ordered set of rules, and is named on the **service-policy** command. In ASDM, the policy map is represented as a folder on the Service Policy Rules page.
2. Rules, each rule being a **class** command within the service policy map and the commands associated with the **class** command. In ASDM, each rule is shown on a separate row, and the name of the rule is the class name.
   a. The **class** command defines the traffic matching criteria for the rule.
   b. The commands associated with class, such as **inspect**, **set connection timeout**, and so forth, define the services and constraints to apply to matching traffic. Note that inspect commands can point to inspection policy maps, which define actions to apply to inspected traffic. Keep in mind that inspection policy maps are not the same as service policy maps.

The following example compares how service policies appear in the CLI with how they appear in ASDM. Note that there is not a one-to-one mapping between the figure call-outs and lines in the CLI.



The following CLI is generated by the rules shown in the figure above.

```
: Access lists used in class maps.
: In ASDM, these map to call-out 3, from the Match to the Time fields.
access-list inside_mpc line 1 extended permit tcp 10.100.10.0 255.255.255.0 any eq sip
access-list inside_mpc_1 line 1 extended deny udp host 10.1.1.15 any eq snmp
access-list inside_mpc_1 line 2 extended permit udp 10.1.1.0 255.255.255.0 any eq snmp
access-list inside_mpc_2 line 1 extended permit icmp any any
: SNMP map for SNMP inspection. Denies all but v3.
: In ASDM, this maps to call-out 4, rule actions, for the class-inside policy.
snmp-map snmp-v3only
  deny version 1
  deny version 2
  deny version 2c
: Inspection policy map to define SIP behavior.
: The sip-high inspection policy map must be referred to by an inspect sip command
: in the service policy map.
: In ASDM, this maps to call-out 4, rule actions, for the sip-class-inside policy.
policy-map type inspect sip sip-high
  parameters
    rtp-conformance enforce-payloadtype
    no traffic-non-sip
    software-version action mask log
```

```
    uri-non-sip action mask log
    state-checking action drop-connection log
    max-forwards-validation action drop log
    strict-header-validation action drop log
: Class map to define traffic matching for the inside-class rule.
: In ASDM, this maps to call-out 3, from the Match to the Time fields.
class-map inside-class
  match access-list inside_mpc_1
: Class map to define traffic matching for the sip-class-inside rule.
: In ASDM, this maps to call-out 3, from the Match to the Time fields.
class-map sip-class-inside
  match access-list inside_mpc
: Class map to define traffic matching for the inside-class1 rule.
: In ASDM, this maps to call-out 3, from the Match to the Time fields.
class-map inside-class1
  match access-list inside_mpc_2
: Policy map that actually defines the service policy rule set named test-inside-policy.
: In ASDM, this corresponds to the folder at call-out 1.
policy-map test-inside-policy
: First rule in test-inside-policy, named sip-class-inside. Inspects SIP traffic.
: The sip-class-inside rule applies the sip-high inspection policy map to SIP inspection.
: In ASDM, each rule corresponds to call-out 2.
  class sip-class-inside
    inspect sip sip-high
: Second rule, inside-class. Applies SNMP inspection using an SNMP map.
  class inside-class
    inspect snmp snmp-v3only
: Third rule, inside-class1. Applies ICMP inspection.
  class inside-class1
    inspect icmp
: Fourth rule, class-default. Applies connection settings and enables user statistics.
  class class-default
    set connection timeout embryonic 0:00:30 half-closed 0:10:00 idle 1:00:00
reset dcd 0:15:00 5
    user-statistics accounting
: The service-policy command applies the policy map rule set to the inside interface.
: This command activates the policies.
service-policy test-inside-policy interface inside
```

Features Configured with Service Policies

The following table lists the features you configure using service policies.

Table 11-1 Features Configured with Service Policies

Feature	For Through Traffic?	For Management Traffic?	See:
Application inspection (multiple types)	**All** except RADIUS accounting	**RADIUS accounting** only	• Chapter 12, "Getting Started with Application Layer Protocol Inspection." • Chapter 13, "Inspection of Basic Internet Protocols." • Chapter 14, "Inspection for Voice and Video Protocols." • Chapter 15, "Inspection of Database, Directory, and Management Protocols." • Chapter 8, "ASA and Cisco Cloud Web Security."
ASA IPS	**Yes**	No	See the ASA IPS quick start guide.
ASA CX	**Yes**	No	See the ASA CX quick start guide.
ASA FirePOWER (ASA SFR)	**Yes**	No	Chapter 7, "ASA FirePOWER Module."
NetFlow Secure Event Logging filtering	**Yes**	**Yes**	See the general operations configuration guide.
QoS input and output policing	**Yes**	No	Chapter 17, "Quality of Service."
QoS standard priority queue	**Yes**	No	Chapter 17, "Quality of Service."
TCP and UDP connection limits and timeouts, and TCP sequence number randomization	**Yes**	**Yes**	Chapter 16, "Connection Settings."
TCP normalization	**Yes**	No	Chapter 16, "Connection Settings."
TCP state bypass	**Yes**	No	Chapter 16, "Connection Settings."
User statistics for Identity Firewall	**Yes**	**Yes**	See the **user-statistics** command in the command reference.

Feature Directionality

Actions are applied to traffic bidirectionally or unidirectionally depending on the feature. For features that are applied bidirectionally, all traffic that enters or exits the interface to which you apply the policy map is affected if the traffic matches the class map for both directions.

Note When you use a global policy, all features are unidirectional; features that are normally bidirectional when applied to a single interface only apply to the ingress of each interface when applied globally. Because the policy is applied to all interfaces, the policy will be applied in both directions so bidirectionality in this case is redundant.

For features that are applied unidirectionally, for example QoS priority queue, only traffic that enters (or exits, depending on the feature) the interface to which you apply the policy map is affected. See the following table for the directionality of each feature.

Table 11-2 Feature Directionality

Feature	Single Interface Direction	Global Direction
Application inspection (multiple types)	Bidirectional	Ingress
ASA CSC	Bidirectional	Ingress
ASA CX	Bidirectional	Ingress
ASA CX authentication proxy	Ingress	Ingress
ASA FirePOWER (ASA SFR)	Bidirectional	Ingress
ASA IPS	Bidirectional	Ingress
NetFlow Secure Event Logging filtering	N/A	Ingress
QoS input policing	Ingress	Ingress
QoS output policing	Egress	Egress
QoS standard priority queue	Egress	Egress
TCP and UDP connection limits and timeouts, and TCP sequence number randomization	Bidirectional	Ingress
TCP normalization	Bidirectional	Ingress
TCP state bypass	Bidirectional	Ingress
User statistics for Identity Firewall	Bidirectional	Ingress

Feature Matching Within a Service Policy

A packet matches class maps in a policy map for a given interface according to the following rules:

1. A packet can match only one class map in the policy map for each feature type.
2. When the packet matches a class map for a feature type, the ASA does not attempt to match it to any subsequent class maps for that feature type.
3. If the packet matches a subsequent class map for a different feature type, however, then the ASA also applies the actions for the subsequent class map, if supported. See Incompatibility of Certain Feature Actions, page 11-6 for more information about unsupported combinations.

Note Application inspection includes multiple inspection types, and most are mutually exclusive. For inspections that can be combined, each inspection is considered to be a separate feature.

Examples of Packet Matching

For example:

- If a packet matches a class map for connection limits, and also matches a class map for an application inspection, then both actions are applied.
- If a packet matches a class map for HTTP inspection, but also matches another class map that includes HTTP inspection, then the second class map actions are not applied.

- If a packet matches a class map for HTTP inspection, but also matches another class map that includes FTP inspection, then the second class map actions are not applied because HTTP and FTP inspections cannot be combined.
- If a packet matches a class map for HTTP inspection, but also matches another class map that includes IPv6 inspection, then both actions are applied because the IPv6 inspection can be combined with any other type of inspection.

Order in Which Multiple Feature Actions are Applied

The order in which different types of actions in a policy map are performed is independent of the order in which the actions appear in the policy map.

Actions are performed in the following order:

1. QoS input policing
2. TCP normalization, TCP and UDP connection limits and timeouts, TCP sequence number randomization, and TCP state bypass.

   Note When a the ASA performs a proxy service (such as AAA or CSC) or it modifies the TCP payload (such as FTP inspection), the TCP normalizer acts in dual mode, where it is applied before and after the proxy or payload modifying service.

3. ASA CSC
4. Application inspections that can be combined with other inspections:
   a. IPv6
   b. IP options
   c. WAAS
5. Application inspections that cannot be combined with other inspections. See Incompatibility of Certain Feature Actions, page 11-6 for more information.
6. ASA IPS
7. ASA CX
8. ASA FirePOWER (ASA SFR)
9. QoS output policing
10. QoS standard priority queue

Note NetFlow Secure Event Logging filtering and User statistics for Identity Firewall are order-independent.

Incompatibility of Certain Feature Actions

Some features are not compatible with each other for the same traffic. The following list might not include all incompatibilities; for information about compatibility of each feature, see the chapter or section for the feature:

- You cannot configure QoS priority queuing and QoS policing for the same set of traffic.

- Most inspections should not be combined with another inspection, so the ASA only applies one inspection if you configure multiple inspections for the same traffic. HTTP inspection can be combined with the Cloud Web Security inspection. Other exceptions are listed in Order in Which Multiple Feature Actions are Applied, page 11-6.
- You cannot configure traffic to be sent to multiple modules, such as the ASA CX and ASA IPS.
- HTTP inspection is not compatible with ASA CX or ASA FirePOWER.
- Cloud Web Security is not compatible with ASA CX or ASA FirePOWER.

Note The **match default-inspection-traffic** command, which is used in the default global policy, is a special CLI shortcut to match the default ports for all inspections. When used in a policy map, this class map ensures that the correct inspection is applied to each packet, based on the destination port of the traffic. For example, when UDP traffic for port 69 reaches the ASA, then the ASA applies the TFTP inspection; when TCP traffic for port 21 arrives, then the ASA applies the FTP inspection. So in this case only, you can configure multiple inspections for the same class map. Normally, the ASA does not use the port number to determine which inspection to apply, thus giving you the flexibility to apply inspections to non-standard ports, for example.

This traffic class does not include the default ports for Cloud Web Security inspection (80 and 443).

An example of a misconfiguration is if you configure multiple inspections in the same policy map and do not use the default-inspection-traffic shortcut. In Example 11-1, traffic destined to port 21 is mistakenly configured for both FTP and HTTP inspection. In Example 11-2, traffic destined to port 80 is mistakenly configured for both FTP and HTTP inspection. In both cases of misconfiguration examples, only the FTP inspection is applied, because FTP comes before HTTP in the order of inspections applied.

Example 11-1 Misconfiguration for FTP packets: HTTP Inspection Also Configured

```
class-map ftp
   match port tcp eq 21
class-map http
   match port tcp eq 21 [it should be 80]
policy-map test
   class ftp
     inspect ftp
   class http
     inspect http
```

Example 11-2 Misconfiguration for HTTP packets: FTP Inspection Also Configured

```
class-map ftp
   match port tcp eq 80 [it should be 21]
class-map http
   match port tcp eq 80
policy-map test
   class ftp
     inspect ftp
   class http
     inspect http
```

Feature Matching for Multiple Service Policies

For TCP and UDP traffic (and ICMP when you enable stateful ICMP inspection), service policies operate on traffic flows, and not just individual packets. If traffic is part of an existing connection that matches a feature in a policy on one interface, that traffic flow cannot also match the same feature in a policy on another interface; only the first policy is used.

For example, if HTTP traffic matches a policy on the inside interface to inspect HTTP traffic, and you have a separate policy on the outside interface for HTTP inspection, then that traffic is not also inspected on the egress of the outside interface. Similarly, the return traffic for that connection will not be inspected by the ingress policy of the outside interface, nor by the egress policy of the inside interface.

For traffic that is not treated as a flow, for example ICMP when you do not enable stateful ICMP inspection, returning traffic can match a different policy map on the returning interface. For example, if you configure IPS on the inside and outside interfaces, but the inside policy uses virtual sensor 1 while the outside policy uses virtual sensor 2, then a non-stateful Ping will match virtual sensor 1 outbound, but will match virtual sensor 2 inbound.

Guidelines for Service Policies

IPv6 Guidelines

Supports IPv6 for the following features:

- Application inspection for DNS, FTP, HTTP, ICMP, ScanSafe, SIP, SMTP, IPsec-pass-thru, and IPv6.
- ASA IPS
- ASA CX
- ASA FirePOWER
- NetFlow Secure Event Logging filtering
- TCP and UDP connection limits and timeouts, TCP sequence number randomization
- TCP normalization
- TCP state bypass
- User statistics for Identity Firewall

Class Map (Traffic Class) Guidelines

The maximum number of class maps (traffic classes) of all types is 255 in single mode or per context in multiple mode. Class maps include the following types:

- Layer 3/4 class maps (for through traffic and management traffic).
- Inspection class maps
- Regular expression class maps
- **match** commands used directly underneath an inspection policy map

This limit also includes default class maps of all types, limiting user-configured class maps to approximately 235. See Default Class Maps (Traffic Classes), page 11-10.

Policy Map Guidelines

See the following guidelines for using policy maps:

- You can only assign one policy map per interface. However you can create up to 64 policy maps in the configuration.
- You can apply the same policy map to multiple interfaces.
- You can identify up to 63 Layer 3/4 class maps in a Layer 3/4 policy map.
- For each class map, you can assign multiple actions from one or more feature types, if supported. See Incompatibility of Certain Feature Actions, page 11-6.

Service Policy Guidelines

- Interface service policies take precedence over the global service policy for a given feature. For example, if you have a global policy with FTP inspection, and an interface policy with TCP normalization, then both FTP inspection and TCP normalization are applied to the interface. However, if you have a global policy with FTP inspection, and an interface policy with FTP inspection, then only the interface policy FTP inspection is applied to that interface.
- You can only apply one global policy. For example, you cannot create a global policy that includes feature set 1, and a separate global policy that includes feature set 2. All features must be included in a single policy.
- When you make service policy changes to the configuration, all *new* connections use the new service policy. Existing connections continue to use the policy that was configured at the time of the connection establishment. Output for the **show** command will not include data about the old connections.

  For example, if you remove a QoS service policy from an interface, then add a modified version, then the **show service-policy** command only displays QoS counters associated with new connections that match the new service policy; existing connections on the old policy no longer show in the command output.

  To ensure that all connections use the new policy, you need to disconnect the current connections so they can reconnect using the new policy. Use the **clear conn** or **clear local-host** commands.

Defaults for Service Policies

The following topics describe the default settings for service policies and the Modular Policy Framework:

- Default Service Policy Configuration, page 11-9
- Default Class Maps (Traffic Classes), page 11-10

Default Service Policy Configuration

By default, the configuration includes a policy that matches all default application inspection traffic and applies certain inspections to the traffic on all interfaces (a global policy). Not all inspections are enabled by default. You can only apply one global policy, so if you want to alter the global policy, you need to either edit the default policy or disable it and apply a new one. (An interface policy overrides the global policy for a particular feature.)

The default policy includes the following application inspections:

- DNS
- FTP

- H323 (H225)
- H323 (RAS)
- RSH
- RTSP
- ESMTP
- SQLnet
- Skinny (SCCP)
- SunRPC
- XDMCP
- SIP
- NetBios
- TFTP
- IP Options

The default policy configuration includes the following commands:

```
class-map inspection_default
 match default-inspection-traffic
policy-map type inspect dns preset_dns_map
 parameters
    message-length maximum client auto
    message-length maximum 512
    dns-guard
    protocol-enforcement
    nat-rewrite
policy-map global_policy
 class inspection_default
  inspect dns preset_dns_map
  inspect ftp
  inspect h323 h225 _default_h323_map
  inspect h323 ras _default_h323_map
  inspect ip-options _default_ip_options_map
  inspect netbios
  inspect rsh
  inspect rtsp
  inspect skinny
  inspect esmtp _default_esmtp_map
  inspect sqlnet
  inspect sunrpc
  inspect tftp
  inspect sip
  inspect xdmcp
service-policy global_policy global
```

Note See Incompatibility of Certain Feature Actions, page 11-6 for more information about the special **match default-inspection-traffic** command used in the default class map.

Default Class Maps (Traffic Classes)

The configuration includes a default Layer 3/4 class map (traffic class) that the ASA uses in the default global policy called default-inspection-traffic; it matches the default inspection traffic. This class, which is used in the default global policy, is a special shortcut to match the default ports for all inspections.

When used in a policy, this class ensures that the correct inspection is applied to each packet, based on the destination port of the traffic. For example, when UDP traffic for port 69 reaches the ASA, then the ASA applies the TFTP inspection; when TCP traffic for port 21 arrives, then the ASA applies the FTP inspection. So in this case only, you can configure multiple inspections for the same class map. Normally, the ASA does not use the port number to determine which inspection to apply, thus giving you the flexibility to apply inspections to non-standard ports, for example.

```
class-map inspection_default
 match default-inspection-traffic
```

Another class map that exists in the default configuration is called class-default, and it matches all traffic. This class map appears at the end of all Layer 3/4 policy maps and essentially tells the ASA to not perform any actions on all other traffic. You can use the class-default class if desired, rather than making your own **match any** class map. In fact, some features are only available for class-default.

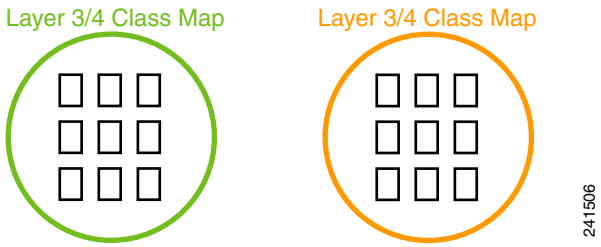
```
class-map class-default
 match any
```

Configure Service Policies

To configure service policies using the Modular Policy Framework, perform the following steps:

Step 1 Identify the traffic on which you want to act by creating Layer 3/4 class maps, as described in Identify Traffic (Layer 3/4 Class Maps), page 11-13.

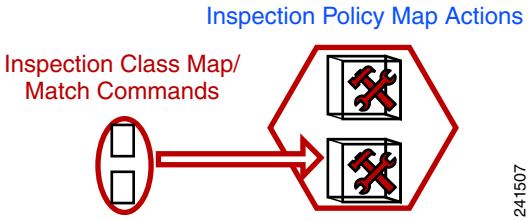
For example, you might want to perform actions on all traffic that passes through the ASA; or you might only want to perform certain actions on traffic from 10.1.1.0/24 to any destination address.



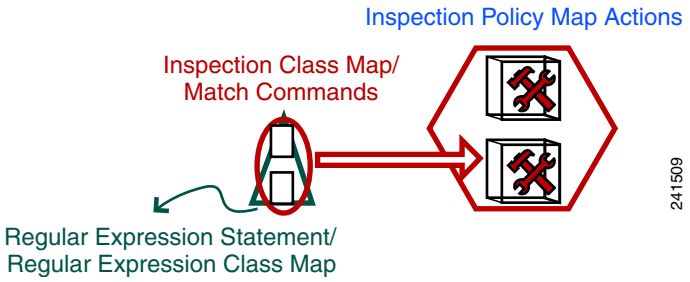
Step 2 Optionally, perform additional actions on some inspection traffic.

If one of the actions you want to perform is application inspection, and you want to perform additional actions on some inspection traffic, then create an inspection policy map. The inspection policy map identifies the traffic and specifies what to do with it.

For example, you might want to drop all HTTP requests with a body length greater than 1000 bytes.

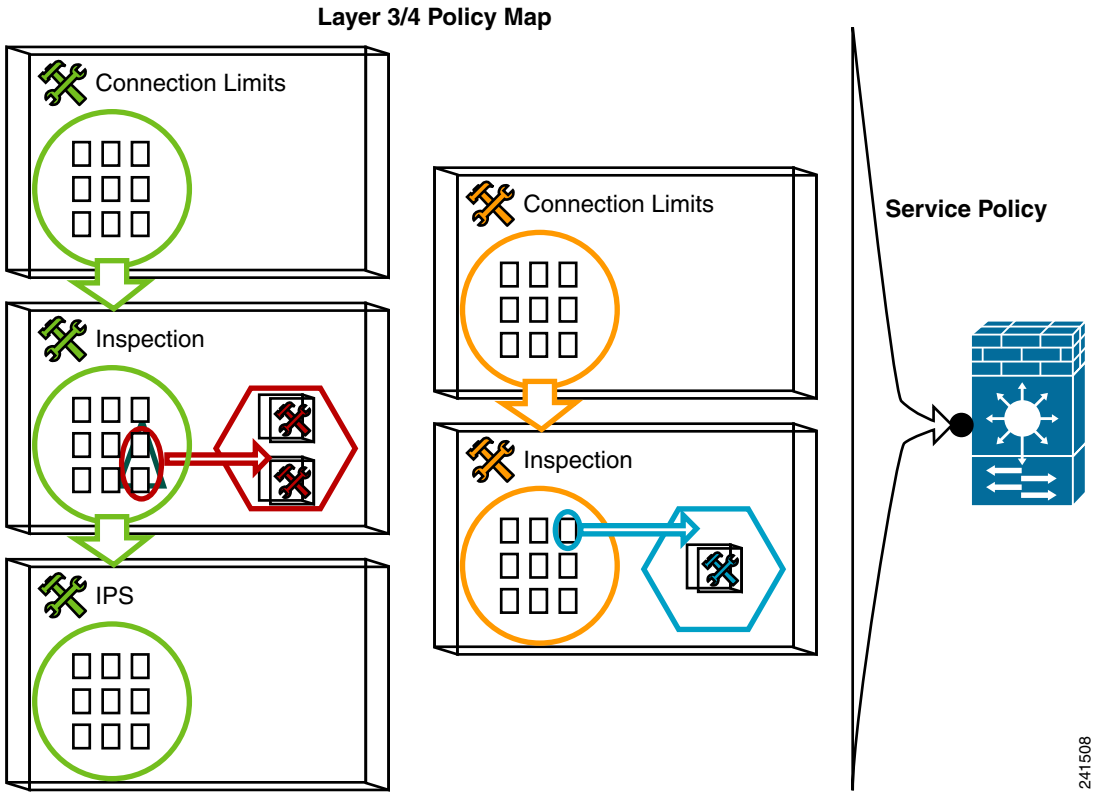


You can create a self-contained inspection policy map that identifies the traffic directly with **match** commands, or you can create an inspection class map for reuse or for more complicated matching. For example, you could match text within a inspected packets using a regular expression or a group of regular expressions (a regular expression class map), and target actions based on narrower criteria. For example, you might want to drop all HTTP requests with a URL including the text "example.com."



See Configure Application Layer Protocol Inspection, page 12-9.

Step 3 Define the actions you want to perform on each Layer 3/4 class map by creating a Layer 3/4 policy map, as described in Define Actions (Layer 3/4 Policy Map), page 11-16.



Step 4 Determine on which interfaces you want to apply the policy map, or apply it globally, as described in Apply Actions to an Interface (Service Policy), page 11-17.

Identify Traffic (Layer 3/4 Class Maps)

A Layer 3/4 class map identifies Layer 3 and 4 traffic to which you want to apply actions. You can create multiple Layer 3/4 class maps for each Layer 3/4 policy map.

- Create a Layer 3/4 Class Map for Through Traffic, page 11-13
- Create a Layer 3/4 Class Map for Management Traffic, page 11-15

Create a Layer 3/4 Class Map for Through Traffic

A Layer 3/4 class map matches traffic based on protocols, ports, IP addresses and other Layer 3 or 4 attributes.

Tip We suggest that you only inspect traffic on ports on which you expect application traffic; if you inspect all traffic, for example using **match any**, the ASA performance can be impacted.

Procedure

Step 1 Create a Layer 3/4 class map, where *class_map_name* is a string up to 40 characters in length.

```
class-map class_map_name
```

The name "class-default" is reserved. All types of class maps use the same name space, so you cannot reuse a name already used by another type of class map. The CLI enters class-map configuration mode.

Example:

```
hostname(config)# class-map all_udp
```

Step 2 (Optional) Add a description to the class map.

```
description string
```

Example:

```
hostname(config-cmap)# description All UDP traffic
```

Step 3 Match traffic using one of the following commands. Unless otherwise specified, you can include only one **match** command in the class map.

- **match any**—Matches all traffic.

  ```
  hostname(config-cmap)# match any
  ```

- **match access-list** *access_list_name*—Matches traffic specified by an extended ACL. If the ASA is operating in transparent firewall mode, you can use an EtherType ACL.

  ```
  hostname(config-cmap)# match access-list udp
  ```

- **match port** {**tcp** | **udp**} {**eq** *port_num* | **range** *port_num port_num*}—Matches TCP or UDP destination ports, either a single port or a contiguous range of ports. For applications that use multiple, non-contiguous ports, use the **match access-list** command and define an ACE to match each port.

  ```
  hostname(config-cmap)# match tcp eq 80
  ```

- **match default-inspection-traffic**—Matches default traffic for inspection: the default TCP and UDP ports used by all applications that the ASA can inspect.

```
hostname(config-cmap)# match default-inspection-traffic
```

This command, which is used in the default global policy, is a special CLI shortcut that when used in a policy map, ensures that the correct inspection is applied to each packet, based on the destination port of the traffic. For example, when UDP traffic for port 69 reaches the ASA, then the ASA applies the TFTP inspection; when TCP traffic for port 21 arrives, then the ASA applies the FTP inspection. So in this case only, you can configure multiple inspections for the same class map (with the exception of WAAS inspection, which can be configured with other inspections. See Incompatibility of Certain Feature Actions, page 11-6 for more information about combining actions). Normally, the ASA does not use the port number to determine the inspection applied, thus giving you the flexibility to apply inspections to non-standard ports, for example.

See Default Inspections and NAT Limitations, page 12-6 for a list of default ports. Not all applications whose ports are included in the **match default-inspection-traffic** command are enabled by default in the policy map.

You can specify a **match access-list** command along with the **match default-inspection-traffic** command to narrow the matched traffic. Because the **match default-inspection-traffic** command specifies the ports and protocols to match, any ports and protocols in the ACL are ignored.

- **match dscp** *value1* [*value2*] [...] [*value8*]—Matches the DSCP value in an IP header, up to eight DSCP values.

  ```
  hostname(config-cmap)# match dscp af43 cs1 ef
  ```

- **match precedence** *value1* [*value2*] [*value3*] [*value4*]—Matches up to four precedence values, represented by the TOS byte in the IP header, where *value1* through *value4* can be 0 to 7, corresponding to the possible precedences.

  ```
  hostname(config-cmap)# match precedence 1 4
  ```

- **match rtp** *starting_port range*—Matches RTP traffic, where the *starting_port* specifies an even-numbered UDP destination port between 2000 and 65534. The *range* specifies the number of additional UDP ports to match above the *starting_port*, between 0 and 16383.

  ```
  hostname(config-cmap)# match rtp 4004 100
  ```

- **match tunnel-group** *name*—Matches VPN tunnel group traffic to which you want to apply QoS.

  You can also specify one other **match** command to refine the traffic match. You can specify any of the preceding commands, except for the **match any**, **match access-list**, or **match default-inspection-traffic** commands. Or you can also enter the **match flow ip destination-address** command to match flows in the tunnel group going to each IP address.

  ```
  hostname(config-cmap)# match tunnel-group group1
  hostname(config-cmap)# match flow ip destination-address
  ```

Examples

The following is an example for the **class-map** command:

```
hostname(config)# access-list udp permit udp any any
hostname(config)# access-list tcp permit tcp any any
hostname(config)# access-list host_foo permit ip any 10.1.1.1 255.255.255.255

hostname(config)# class-map all_udp
hostname(config-cmap)# description "This class-map matches all UDP traffic"
hostname(config-cmap)# match access-list udp

hostname(config-cmap)# class-map all_tcp
```

```
hostname(config-cmap)# description "This class-map matches all TCP traffic"
hostname(config-cmap)# match access-list tcp

hostname(config-cmap)# class-map all_http
hostname(config-cmap)# description "This class-map matches all HTTP traffic"
hostname(config-cmap)# match port tcp eq http

hostname(config-cmap)# class-map to_server
hostname(config-cmap)# description "This class-map matches all traffic to server 10.1.1.1"
hostname(config-cmap)# match access-list host_foo
```

Create a Layer 3/4 Class Map for Management Traffic

For management traffic to the ASA, you might want to perform actions specific to this kind of traffic. You can specify a management class map that can match an ACL or TCP or UDP ports. The types of actions available for a management class map in the policy map are specialized for management traffic. See Features Configured with Service Policies, page 11-4.

Procedure

Step 1 Create a management class map, where *class_map_name* is a string up to 40 characters in length.

class-map type management *class_map_name*

The name “class-default” is reserved. All types of class maps use the same name space, so you cannot reuse a name already used by another type of class map. The CLI enters class-map configuration mode.

Example:

```
hostname(config)# class-map management all_udp
```

Step 2 (Optional) Add a description to the class map.

description *string*

Example:

```
hostname(config-cmap)# description All UDP traffic
```

Step 3 Match traffic using one of the following commands.

- **match access-list** *access_list_name*—Matches traffic specified by an extended ACL. If the ASA is operating in transparent firewall mode, you can use an EtherType ACL.

  ```
  hostname(config-cmap)# match access-list udp
  ```

- **match port** {**tcp** | **udp**} {**eq** *port_num* | **range** *port_num port_num*}—Matches TCP or UDP destination ports, either a single port or a contiguous range of ports. For applications that use multiple, non-contiguous ports, use the **match access-list** command and define an ACE to match each port.

  ```
  hostname(config-cmap)# match tcp eq 80
  ```

Define Actions (Layer 3/4 Policy Map)

After you configure Layer 3/4 class maps to identify traffic, use a Layer 3/4 policy map to associate actions to those classes.

Tip The maximum number of policy maps is 64, but you can only apply one policy map per interface.

Procedure

Step 1 Add the policy map: **policy-map** *policy_map_name*

Where *policy_map_name* argument is the name of the policy map, up to 40 characters in length. All types of policy maps use the same name space, so you cannot reuse a name already used by another type of policy map. The CLI enters policy-map configuration mode.

Example:

```
hostname(config)# policy-map global_policy
```

Step 2 Specify a previously configured Layer 3/4 class map: **class** *class_map_name*

Where the *class_map_name* is the name of the class map.

See Identify Traffic (Layer 3/4 Class Maps), page 11-13 to add a class map.

Example:

```
hostname(config-pmap)# class all-http
```

Step 3 Specify one or more actions for this class map.

See Features Configured with Service Policies, page 11-4.

Note If there is no **match default-inspection-traffic** command in a class map, then at most one **inspect** command is allowed to be configured under the class.

Step 4 Repeat the process for each class map you want to include in this policy map.

Examples

The following is an example of a **policy-map** command for a connection policy. It limits the number of connections allowed to the web server 10.1.1.1:

```
hostname(config)# access-list http-server permit tcp any host 10.1.1.1
hostname(config)# class-map http-server
hostname(config-cmap)# match access-list http-server

hostname(config)# policy-map global-policy
hostname(config-pmap)# description This policy map defines a policy concerning connection
to http server.
hostname(config-pmap)# class http-server
hostname(config-pmap-c)# set connection conn-max 256
```

The following example shows how multi-match works in a policy map:

```
hostname(config)# class-map inspection_default
hostname(config-cmap)# match default-inspection-traffic
hostname(config)# class-map http_traffic
hostname(config-cmap)# match port tcp eq 80
```

```
hostname(config)# policy-map outside_policy
hostname(config-pmap)# class inspection_default
hostname(config-pmap-c)# inspect http http_map
hostname(config-pmap-c)# inspect sip
hostname(config-pmap)# class http_traffic
hostname(config-pmap-c)# set connection timeout idle 0:10:0
```

The following example shows how traffic matches the first available class map, and will not match any subsequent class maps that specify actions in the same feature domain:

```
hostname(config)# class-map telnet_traffic
hostname(config-cmap)# match port tcp eq 23
hostname(config)# class-map ftp_traffic
hostname(config-cmap)# match port tcp eq 21
hostname(config)# class-map tcp_traffic
hostname(config-cmap)# match port tcp range 1 65535
hostname(config)# class-map udp_traffic
hostname(config-cmap)# match port udp range 0 65535
hostname(config)# policy-map global_policy
hostname(config-pmap)# class telnet_traffic
hostname(config-pmap-c)# set connection timeout idle 0:0:0
hostname(config-pmap-c)# set connection conn-max 100
hostname(config-pmap)# class ftp_traffic
hostname(config-pmap-c)# set connection timeout idle 0:5:0
hostname(config-pmap-c)# set connection conn-max 50
hostname(config-pmap)# class tcp_traffic
hostname(config-pmap-c)# set connection timeout idle 2:0:0
hostname(config-pmap-c)# set connection conn-max 2000
```

When a Telnet connection is initiated, it matches **class telnet_traffic**. Similarly, if an FTP connection is initiated, it matches **class ftp_traffic**. For any TCP connection other than Telnet and FTP, it will match **class tcp_traffic**. Even though a Telnet or FTP connection can match **class tcp_traffic**, the ASA does not make this match because they previously matched other classes.

Apply Actions to an Interface (Service Policy)

To activate the Layer 3/4 policy map, create a service policy that applies it to one or more interfaces or that applies it globally to all interfaces. Use the following command:

service-policy *policy_map_name* {**global** | **interface** *interface_name*} [**fail-close**]

Where:

- *policy_map_name* is the name of the policy map.
- **global** creates a service policy that applies to all interfaces that do not have a specific policy.

  You can only apply one global policy, so if you want to alter the global policy, you need to either edit the default policy or disable it and apply a new one. By default, the configuration includes a global policy that matches all default application inspection traffic and applies inspection to the traffic globally. The default service policy includes the following command: **service-policy global_policy global**.
- **interface** *interface_name* creates a service policy by associating a policy map with an interface.
- **fail-close** generates a syslog (767001) for IPv6 traffic that is dropped by application inspections that do not support IPv6 traffic. By default, syslogs are not generated.

Examples

For example, the following command enables the inbound_policy policy map on the outside interface:

```
hostname(config)# service-policy inbound_policy interface outside
```

The following commands disable the default global policy, and enables a new one called new_global_policy.

```
hostname(config)# no service-policy global_policy global
hostname(config)# service-policy new_global_policy global
```

Monitoring Service Policies

To monitor service policies, enter the following command:

- **show service-policy**

  Displays the service policy statistics.

Examples for Service Policies (Modular Policy Framework)

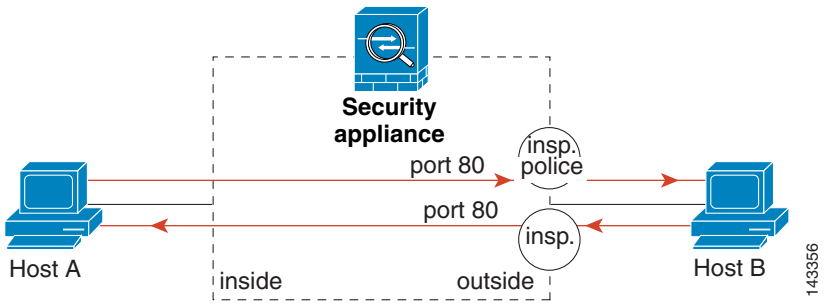
This section includes several Modular Policy Framework examples.

- Applying Inspection and QoS Policing to HTTP Traffic, page 11-18
- Applying Inspection to HTTP Traffic Globally, page 11-19
- Applying Inspection and Connection Limits to HTTP Traffic to Specific Servers, page 11-19
- Applying Inspection to HTTP Traffic with NAT, page 11-20

Applying Inspection and QoS Policing to HTTP Traffic

In this example, any HTTP connection (TCP traffic on port 80) that enters or exits the ASA through the outside interface is classified for HTTP inspection. Any HTTP traffic that exits the outside interface is classified for policing.

Figure 11-1 HTTP Inspection and QoS Policing



See the following commands for this example:

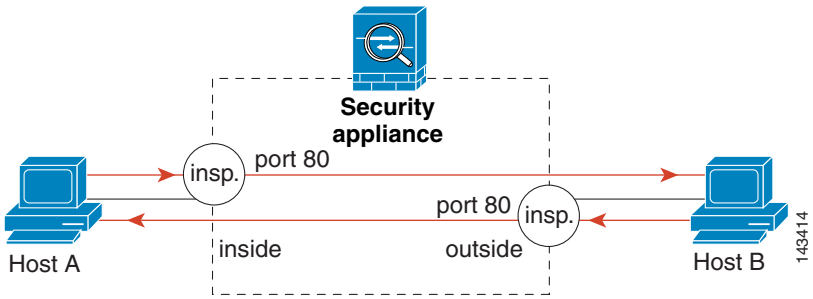
```
hostname(config)# class-map http_traffic
```

```
hostname(config-cmap)# match port tcp eq 80

hostname(config)# policy-map http_traffic_policy
hostname(config-pmap)# class http_traffic
hostname(config-pmap-c)# inspect http
hostname(config-pmap-c)# police output 250000
hostname(config)# service-policy http_traffic_policy interface outside
```

Applying Inspection to HTTP Traffic Globally

In this example, any HTTP connection (TCP traffic on port 80) that enters the ASA through any interface is classified for HTTP inspection. Because the policy is a global policy, inspection occurs only as the traffic enters each interface.

Figure 11-2 Global HTTP Inspection



See the following commands for this example:

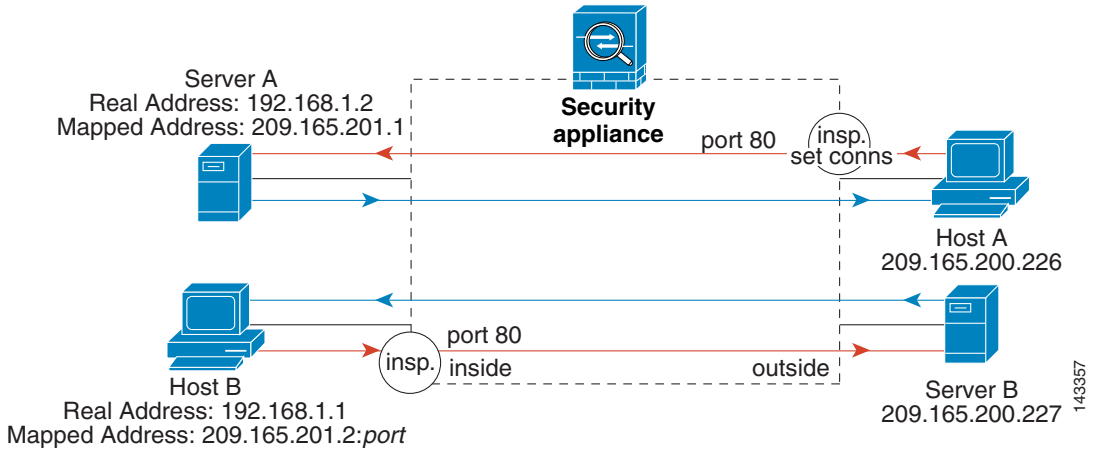
```
hostname(config)# class-map http_traffic
hostname(config-cmap)# match port tcp eq 80

hostname(config)# policy-map http_traffic_policy
hostname(config-pmap)# class http_traffic
hostname(config-pmap-c)# inspect http
hostname(config)# service-policy http_traffic_policy global
```

Applying Inspection and Connection Limits to HTTP Traffic to Specific Servers

In this example, any HTTP connection destined for Server A (TCP traffic on port 80) that enters the ASA through the outside interface is classified for HTTP inspection and maximum connection limits. Connections initiated from Server A to Host A do not match the ACL in the class map, so they are not affected.

Any HTTP connection destined for Server B that enters the ASA through the inside interface is classified for HTTP inspection. Connections initiated from Server B to Host B do not match the ACL in the class map, so they are not affected.

Figure 11-3 HTTP Inspection and Connection Limits to Specific Servers



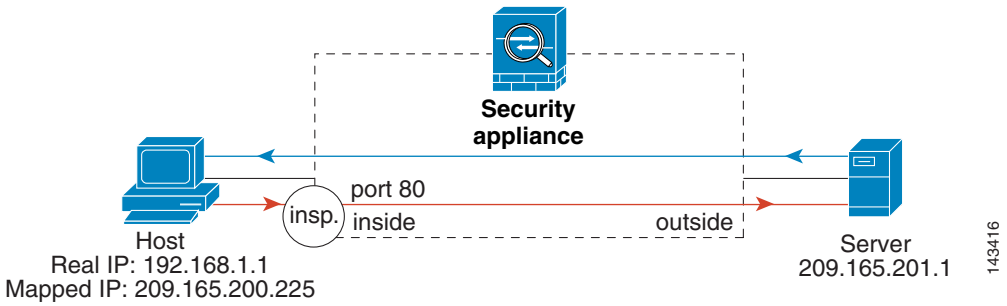
See the following commands for this example:

```
hostname(config)# object network obj-192.168.1.2
hostname(config-network-object)# host 192.168.1.2
hostname(config-network-object)# nat (inside,outside) static 209.165.201.1
hostname(config)# object network obj-192.168.1.0
hostname(config-network-object)# subnet 192.168.1.0 255.255.255.0
hostname(config-network-object)# nat (inside,outside) dynamic 209.165.201.2
hostname(config)# access-list serverA extended permit tcp any host 209.165.201.1 eq 80
hostname(config)# access-list ServerB extended permit tcp any host 209.165.200.227 eq 80

hostname(config)# class-map http_serverA
hostname(config-cmap)# match access-list serverA
hostname(config)# class-map http_serverB
hostname(config-cmap)# match access-list serverB

hostname(config)# policy-map policy_serverA
hostname(config-pmap)# class http_serverA
hostname(config-pmap-c)# inspect http
hostname(config-pmap-c)# set connection conn-max 100
hostname(config)# policy-map policy_serverB
hostname(config-pmap)# class http_serverB
hostname(config-pmap-c)# inspect http

hostname(config)# service-policy policy_serverB interface inside
hostname(config)# service-policy policy_serverA interface outside
```

Applying Inspection to HTTP Traffic with NAT

In this example, the Host on the inside network has two addresses: one is the real IP address 192.168.1.1, and the other is a mapped IP address used on the outside network, 209.165.200.225. You must use the real IP address in the ACL in the class map. If you applied it to the outside interface, you would also use the real address.

Figure 11-4 HTTP Inspection with NAT



See the following commands for this example:

```
hostname(config)# object network obj-192.168.1.1
hostname(config-network-object)# host 192.168.1.1
hostname(config-network-object)# nat (VM1,outside) static 209.165.200.225

hostname(config)# access-list http_client extended permit tcp host 192.168.1.1 any eq 80

hostname(config)# class-map http_client
hostname(config-cmap)# match access-list http_client

hostname(config)# policy-map http_client
hostname(config-pmap)# class http_client
hostname(config-pmap-c)# inspect http

hostname(config)# service-policy http_client interface inside
```

History for Service Policies

Feature Name	Releases	Description
Modular Policy Framework	7.0(1)	Modular Policy Framework was introduced.
Management class map for use with RADIUS accounting traffic	7.2(1)	The management class map was introduced for use with RADIUS accounting traffic. The following commands were introduced: **class-map type management**, and **inspect radius-accounting**.
Inspection policy maps	7.2(1)	The inspection policy map was introduced. The following command was introduced: **class-map type inspect**.
Regular expressions and policy maps	7.2(1)	Regular expressions and policy maps were introduced to be used under inspection policy maps. The following commands were introduced: **class-map type regex**, **regex**, **match regex**.
Match any for inspection policy maps	8.0(2)	The **match any** keyword was introduced for use with inspection policy maps: traffic can match one or more criteria to match the class map. Formerly, only **match all** was available.

Getting Started with Application Layer Protocol Inspection

The following topics describe how to configure application layer protocol inspection.

Application Layer Protocol Inspection

Inspection engines are required for services that embed IP addressing information in the user data packet or that open secondary channels on dynamically assigned ports. These protocols require the ASA to do a deep packet inspection instead of passing the packet through the fast path (see the general operations configuration guide for more information about the fast path). As a result, inspection engines can affect overall throughput. Several common inspection engines are enabled on the ASA by default, but you might need to enable others depending on your network.

The following topics explain application inspection in more detail.

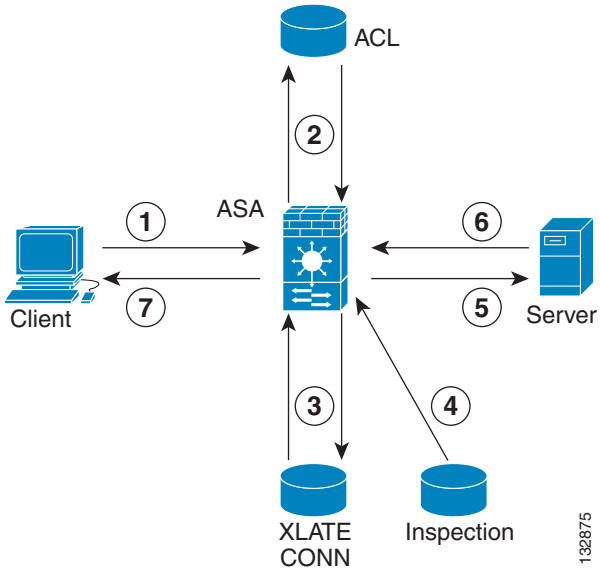
How Inspection Engines Work

As illustrated in the following figure, the ASA uses three databases for its basic operation:

- ACLs—Used for authentication and authorization of connections based on specific networks, hosts, and services (TCP/UDP port numbers).
- Inspections—Contains a static, predefined set of application-level inspection functions.

- Connections (XLATE and CONN tables)—Maintains state and other information about each established connection. This information is used by the Adaptive Security Algorithm and cut-through proxy to efficiently forward traffic within established sessions.

Figure 12-1 How Inspection Engines Work



In this figure, operations are numbered in the order they occur:

1. A TCP SYN packet arrives at the ASA to establish a new connection.
2. The ASA checks the ACL database to determine if the connection is permitted.
3. The ASA creates a new entry in the connection database (XLATE and CONN tables).
4. The ASA checks the Inspections database to determine if the connection requires application-level inspection.
5. After the application inspection engine completes any required operations for the packet, the ASA forwards the packet to the destination system.
6. The destination system responds to the initial request.
7. The ASA receives the reply packet, looks up the connection in the connection database, and forwards the packet because it belongs to an established session.

The default configuration of the ASA includes a set of application inspection entries that associate supported protocols with specific TCP or UDP port numbers and that identify any special handling required.

When to Use Application Protocol Inspection

When a user establishes a connection, the ASA checks the packet against ACLs, creates an address translation, and creates an entry for the session in the fast path, so that further packets can bypass time-consuming checks. However, the fast path relies on predictable port numbers and does not perform address translations inside a packet.

Many protocols open secondary TCP or UDP ports. The initial session on a well-known port is used to negotiate dynamically assigned port numbers.

Other applications embed an IP address in the packet that needs to match the source address that is normally translated when it goes through the ASA.

If you use applications like these, then you need to enable application inspection.

When you enable application inspection for a service that embeds IP addresses, the ASA translates embedded addresses and updates any checksum or other fields that are affected by the translation.

When you enable application inspection for a service that uses dynamically assigned ports, the ASA monitors sessions to identify the dynamic port assignments, and permits data exchange on these ports for the duration of the specific session.

Inspection Policy Maps

You can configure special actions for many application inspections using an *inspection policy map*. These maps are optional: you can enable inspection for a protocol that supports inspection policy maps without configuring a map. These maps are needed only if you want something other than the default inspection actions.

See Configure Application Layer Protocol Inspection, page 12-9 for a list of applications that support inspection policy maps.

An inspection policy map consists of one or more of the following elements. The exact options available for an inspection policy map depends on the application.

- Traffic matching criteria—You match application traffic to criteria specific to the application, such as a URL string, for which you then enable actions.

  For some traffic matching criteria, you use regular expressions to match text inside a packet. Be sure to create and test the regular expressions before you configure the policy map, either singly or grouped together in a regular expression class map.

- Inspection class map—Some inspection policy maps let you use an inspection class map to include multiple traffic matching criteria. You then identify the inspection class map in the inspection policy map and enable actions for the class as a whole. The difference between creating a class map and defining the traffic match directly in the inspection policy map is that you can create more complex match criteria and you can reuse class maps. However, you cannot set different actions for different matches.

- Parameters—Parameters affect the behavior of the inspection engine.

The following topics provide more details:

- Replacing an In-Use Inspection Policy Map, page 12-3
- How Multiple Traffic Classes are Handled, page 12-4

Replacing an In-Use Inspection Policy Map

If you need to replace an inspection policy map that you are already using in a service policy, use the following methods:

- All inspection policy maps—If you want to exchange an in-use inspection policy map for a different map name, you must remove the **inspect** *protocol map* command, and add it back with the new map. For example:

```
hostname(config)# policy-map test
hostname(config-pmap)# class sip
hostname(config-pmap-c)# no inspect sip sip-map1
hostname(config-pmap-c)# inspect sip sip-map2
```

- HTTP inspection policy maps—If you modify an in-use HTTP inspection policy map (**policy-map type inspect http**), you must remove and reapply the **inspect http** *map* action for the changes to take effect. For example, if you modify the "http-map" inspection policy map, you must remove **inspect http http-map** command from the layer 3/4 policy, then add it back:

```
hostname(config)# policy-map test
hostname(config-pmap)# class http
hostname(config-pmap-c)# no inspect http http-map
hostname(config-pmap-c)# inspect http http-map
```

How Multiple Traffic Classes are Handled

You can specify multiple inspection class maps or direct matches in the inspection policy map.

If a packet matches multiple different **match** or **class** commands, then the order in which the ASA applies the actions is determined by internal ASA rules, and not by the order they are added to the inspection policy map. The internal rules are determined by the application type and the logical progression of parsing a packet, and are not user-configurable. For example for HTTP traffic, parsing a Request Method field precedes parsing the Header Host Length field; an action for the Request Method field occurs before the action for the Header Host Length field. For example, the following match commands can be entered in any order, but the **match request method get** command is matched first.

```
match request header host length gt 100
  reset
match request method get
  log
```

If an action drops a packet, then no further actions are performed in the inspection policy map. For example, if the first action is to reset the connection, then it will never match any further match criteria. If the first action is to log the packet, then a second action, such as resetting the connection, can occur.

If a packet matches multiple **match** or **class** commands that are the same, then they are matched in the order they appear in the policy map. For example, for a packet with the header length of 1001, it will match the first command below, and be logged, and then will match the second command and be reset. If you reverse the order of the two **match** commands, then the packet will be dropped and the connection reset before it can match the second **match** command; it will never be logged.

```
match request header length gt 100
  log
match request header length gt 1000
  reset
```

A class map is determined to be the same type as another class map or **match** command based on the lowest priority **match** command in the class map (the priority is based on the internal rules). If a class map has the same type of lowest priority **match** command as another class map, then the class maps are matched according to the order they are added to the policy map. If the lowest priority match for each class map is different, then the class map with the higher priority **match** command is matched first. For example, the following three class maps contain two types of **match** commands: **match request-cmd** (higher priority) and **match filename** (lower priority). The ftp3 class map includes both commands, but it is ranked according to the lowest priority command, **match filename**. The ftp1 class map includes the highest priority command, so it is matched first, regardless of the order in the policy map. The ftp3 class map is ranked as being of the same priority as the ftp2 class map, which also contains the **match filename** command. They are matched according to the order in the policy map: ftp3 and then ftp2.

```
class-map type inspect ftp match-all ftp1
  match request-cmd get
class-map type inspect ftp match-all ftp2
```

```
  match filename regex abc
class-map type inspect ftp match-all ftp3
  match request-cmd get
  match filename regex abc

policy-map type inspect ftp ftp
  class ftp3
    log
  class ftp2
    log
  class ftp1
    log
```

Guidelines for Application Inspection

Failover

State information for multimedia sessions that require inspection are not passed over the state link for stateful failover. The exceptions are GTP and SIP, which are replicated over the state link.

IPv6

Supports IPv6 for the following inspections:

- DNS
- FTP
- HTTP
- ICMP
- SCCP (Skinny)
- SIP
- SMTP
- IPsec pass-through
- IPv6
- VXLAN

Supports NAT64 for the following inspections:

- DNS
- FTP
- HTTP
- ICMP

Additional Guidelines

- Some inspection engines do not support PAT, NAT, outside NAT, or NAT between same security interfaces. For more information about NAT support, see Default Inspections and NAT Limitations, page 12-6.
- For all the application inspections, the ASA limits the number of simultaneous, active data connections to 200 connections. For example, if an FTP client opens multiple secondary connections, the FTP inspection engine allows only 200 active connections and the 201 connection is dropped and the adaptive security appliance generates a system error message.

- Inspected protocols are subject to advanced TCP-state tracking, and the TCP state of these connections is not automatically replicated. While these connections are replicated to the standby unit, there is a best-effort attempt to re-establish a TCP state.
- TCP/UDP Traffic directed to the ASA (to an interface) is inspected by default. However, ICMP traffic directed to an interface is never inspected, even if you enable ICMP inspection. Thus, a ping (echo request) to an interface can fail under specific circumstances, such as when the echo request comes from a source that the ASA can reach through a backup default route.

Defaults for Application Inspection

The following topics explain the default operations for application inspection.

- Default Inspections and NAT Limitations, page 12-6
- Default Inspection Policy Maps, page 12-9

Default Inspections and NAT Limitations

By default, the configuration includes a policy that matches all default application inspection traffic and applies inspection to the traffic on all interfaces (a global policy). Default application inspection traffic includes traffic to the default ports for each protocol. You can only apply one global policy, so if you want to alter the global policy, for example, to apply inspection to non-standard ports, or to add inspections that are not enabled by default, you need to either edit the default policy or disable it and apply a new one.

The following table lists all inspections supported, the default ports used in the default class map, and the inspection engines that are on by default, shown in bold. This table also notes any NAT limitations. In this table:

- Inspection engines that are enabled by default for the default port are in bold.
- The ASA is in compliance with the indicated standards, but it does not enforce compliance on packets being inspected. For example, FTP commands are supposed to be in a particular order, but the ASA does not enforce the order.

Table 12-1 Supported Application Inspection Engines

Application	Default Port	NAT Limitations	Standards	Comments
CTIQBE	TCP/2748	No extended PAT. No NAT64. (Clustering) No static PAT.	—	—
DCERPC	TCP/135	No NAT64.	—	—
DNS over UDP	UDP/53	No NAT support is available for name resolution through WINS.	RFC 1123	—
FTP	TCP/21	(Clustering) No static PAT.	RFC 959	—
GTP	UDP/3386 UDP/2123	No extended PAT. No NAT.	—	Requires a special license.

Table 12-1 Supported Application Inspection Engines (continued)

Application	Default Port	NAT Limitations	Standards	Comments
H.323 H.225 and **RAS**	TCP/1720 UDP/1718 UDP (RAS) 1718-1719	No dynamic NAT or PAT. Static PAT may not work. (Clustering) No static PAT. No extended PAT. No per-session PAT. No NAT on same security interfaces. No NAT64.	ITU-T H.323, H.245, H225.0, Q.931, Q.932	—
HTTP	TCP/80	—	RFC 2616	Beware of MTU limitations stripping ActiveX and Java. If the MTU is too small to allow the Java or ActiveX tag to be included in one packet, stripping may not occur.
ICMP	—	—	—	ICMP traffic directed to an ASA interface is never inspected.
ICMP ERROR	—	—	—	—
ILS (LDAP)	TCP/389	No extended PAT. No NAT64.	—	—
Instant Messaging (IM)	Varies by client	No extended PAT. No NAT64.	RFC 3860	—
IP Options	—	No NAT64.	RFC 791, RFC 2113	—
IPsec Pass Through	UDP/500	No PAT. No NAT64.	—	—
IPv6	—	No NAT64.	RFC 2460	—
MGCP	UDP/2427, 2727	No extended PAT. No NAT64. (Clustering) No static PAT.	RFC 2705bis-05	—
MMP	TCP 5443	No extended PAT. No NAT64.	—	—
NetBIOS Name Server over IP	UDP/137, 138 (Source ports)	No extended PAT. No NAT64.	—	NetBIOS is supported by performing NAT of the packets for NBNS UDP port 137 and NBDS UDP port 138.
PPTP	TCP/1723	No NAT64. (Clustering) No static PAT.	RFC 2637	—
RADIUS Accounting	1646	No NAT64.	RFC 2865	—

Table 12-1 Supported Application Inspection Engines (continued)

Application	Default Port	NAT Limitations	Standards	Comments
RSH	TCP/514	No PAT. No NAT64. (Clustering) No static PAT.	Berkeley UNIX	—
RTSP	TCP/554	No extended PAT. No NAT64. (Clustering) No static PAT.	RFC 2326, 2327, 1889	No handling for HTTP cloaking.
ScanSafe (Cloud Web Security)	TCP/80 TCP/413	—	—	These ports are not included in the default-inspection-traffic class for the ScanSafe inspection.
SIP	TCP/5060 UDP/5060	No NAT on same security interfaces. No extended PAT. No per-session PAT. No NAT64 or NAT46. (Clustering) No static PAT.	RFC 2543	Does not handle TFTP uploaded Cisco IP Phone configurations under certain circumstances.
SKINNY (SCCP)	TCP/2000	No NAT on same security interfaces. No extended PAT. No per-session PAT. No NAT64, NAT46, or NAT66. (Clustering) No static PAT.	—	Does not handle TFTP uploaded Cisco IP Phone configurations under certain circumstances.
SMTP and **ESMTP**	TCP/25	No NAT64.	RFC 821, 1123	—
SNMP	UDP/161, 162	No NAT or PAT.	RFC 1155, 1157, 1212, 1213, 1215	v.2 RFC 1902-1908; v.3 RFC 2570-2580.
SQL*Net	TCP/1521	No extended PAT. No NAT64. (Clustering) No static PAT.	—	v.1 and v.2.
Sun RPC over UDP and TCP	UDP/111	No extended PAT. No NAT64.	—	The default rule includes UDP port 111; if you want to enable Sun RPC inspection for TCP port 111, you need to create a new rule that matches TCP port 111 and performs Sun RPC inspection.
TFTP	UDP/69	No NAT64. (Clustering) No static PAT.	RFC 1350	Payload IP addresses are not translated.
WAAS	TCP/1-65535	No extended PAT. No NAT64.	—	—

Table 12-1 Supported Application Inspection Engines (continued)

Application	Default Port	NAT Limitations	Standards	Comments
XDMCP	UDP/177	No extended PAT. No NAT64. (Clustering) No static PAT.	—	—
VXLAN	UDP/4789	Not applicable	RFC 7348	Virtual Extensible Local Area Network.

The default policy configuration includes the following commands:

```
class-map inspection_default
 match default-inspection-traffic
policy-map type inspect dns preset_dns_map
 parameters
    message-length maximum client auto
    message-length maximum 512
    dns-guard
    protocol-enforcement
    nat-rewrite
policy-map global_policy
 class inspection_default
  inspect dns preset_dns_map
  inspect ftp
  inspect h323 h225 _default_h323_map
  inspect h323 ras _default_h323_map
  inspect ip-options _default_ip_options_map
  inspect netbios
  inspect rsh
  inspect rtsp
  inspect skinny
  inspect esmtp _default_esmtp_map
  inspect sqlnet
  inspect sunrpc
  inspect tftp
  inspect sip
  inspect xdmcp
```

Default Inspection Policy Maps

Some inspection types use hidden default policy maps. For example, if you enable ESMTP inspection without specifying a map, _default_esmtp_map is used.

The default inspection is described in the sections that explain each inspection type. You can view these default maps using the **show running-config all policy-map** command.

DNS inspection is the only one that uses an explicitly-configured default map, preset_dns_map.

Configure Application Layer Protocol Inspection

You configure application inspection in service policies. Service policies provide a consistent and flexible way to configure ASA features. For example, you can use a service policy to create a timeout configuration that is specific to a particular TCP application, as opposed to one that applies to all TCP applications. For some applications, you can perform special actions when you enable inspection. See

Service Policy Using the Modular Policy Framework, page 11-1 for information about service policies in general.

Inspection is enabled by default for some applications. See Default Inspections and NAT Limitations, page 12-6 section for more information. Use this section to modify your inspection policy.

Procedure

Step 1 Unless you are adding inspection to an existing class map, identify the traffic to which you want to apply inspections in a Layer 3/4 class map either for through traffic or for management traffic.

See Create a Layer 3/4 Class Map for Through Traffic, page 11-13 and Create a Layer 3/4 Class Map for Management Traffic, page 11-15 for detailed information. The management Layer 3/4 class map can be used only with the RADIUS accounting inspection.

There are important implications for the class map that you choose. You can have more than one inspection on the inspection_default class only, and you might want to simply edit the existing global policy that applies the inspection defaults. For detailed information on which class map to choose, see Choosing the Right Traffic Class for Inspection, page 12-14.

Step 2 (Optional) Some inspection engines let you control additional parameters when you apply the inspection to the traffic. The table later in this procedure shows which protocols allow inspection policy maps, with pointers to the instructions on configuring them.

Step 3 Add or edit a Layer 3/4 policy map that sets the actions to take with the class map traffic.

```
hostname(config)# policy-map name
hostname(config-pmap)#
```

The default policy map is called “global_policy.” This policy map includes the default inspections listed in Default Inspections and NAT Limitations, page 12-6. If you want to modify the default policy (for example, to add or delete an inspection, or to identify an additional class map for your actions), then enter **global_policy** as the name.

Step 4 Identify the class map to which you want to assign an action.

```
hostname(config-pmap)# class class_map_name
hostname(config-pmap-c)#
```

If you are editing the default policy map, it includes the inspection_default class map. You can edit the actions for this class by entering **inspection_default** as the name. To add an additional class map to this policy map, identify a different name.

You can combine multiple class maps in the same policy if desired, so you can create one class map to match certain traffic, and another to match different traffic. However, if traffic matches a class map that contains an inspection command, and then matches another class map that also has an inspection command, only the first matching class is used. For example, SNMP matches the inspection_default class map. To enable SNMP inspection, enable SNMP inspection for the default class. Do not add another class that matches SNMP.

Step 5 Enable application inspection.

```
hostname(config-pmap-c)# inspect protocol
```

The *protocol* is one of the following values:

Table 12-2 Protocol Keywords

Keywords	Notes
ctiqbe	See CTIQBE Inspection, page 14-1.
dcerpc [*map_name*]	See DCERPC Inspection, page 15-1. If you added a DCERPC inspection policy map according to Configure a DCERPC Inspection Policy Map, page 15-2, identify the map name in this command.
dns [*map_name*] [**dynamic-filter-snoop**]	See DNS Inspection, page 13-1. If you added a DNS inspection policy map according to Configure DNS Inspection Policy Map, page 13-3, identify the map name in this command. The default DNS inspection policy map name is "preset_dns_map." To enable DNS snooping for the Botnet Traffic Filter, enter the **dynamic-filter-snoop** keyword.
esmtp [*map_name*]	See SMTP and Extended SMTP Inspection, page 13-39. If you added an ESMTP inspection policy map according to Configure an ESMTP Inspection Policy Map, page 13-42, identify the map name in this command.
ftp [**strict** [*map_name*]]	See FTP Inspection, page 13-8. Use the **strict** keyword to increase the security of protected networks by preventing web browsers from sending embedded commands in FTP requests. See Strict FTP, page 13-9 for more information. If you added an FTP inspection policy map according to Configure an FTP Inspection Policy Map, page 13-10, identify the map name in this command.
gtp [*map_name*]	See GTP Inspection, page 15-5. If you added a GTP inspection policy map according to Configure a GTP Inspection Policy Map, page 15-7, identify the map name in this command.
h323 h225 [*map_name*]	See H.323 Inspection, page 14-3. If you added an H323 inspection policy map according to Configure H.323 Inspection Policy Map, page 14-6, identify the map name in this command.
h323 ras [*map_name*]	See H.323 Inspection, page 14-3. If you added an H323 inspection policy map according to Configure H.323 Inspection Policy Map, page 14-6, identify the map name in this command.
http [*map_name*]	See HTTP Inspection, page 13-14. If you added an HTTP inspection policy map according to Configure an HTTP Inspection Policy Map, page 13-16, identify the map name in this command.
icmp	See ICMP Inspection, page 13-21.

Table 12-2 Protocol Keywords

Keywords	Notes
icmp error	See ICMP Error Inspection, page 13-21.
ils	See ILS Inspection, page 15-12.
im [*map_name*]	See Instant Messaging Inspection, page 13-21. If you added an Instant Messaging inspection policy map according to Configure an Instant Messaging Inspection Policy Map, page 13-22, identify the map name in this command.
ip-options [*map_name*]	See IP Options Inspection, page 13-26. If you added an IP Options inspection policy map according to Configure an IP Options Inspection Policy Map, page 13-28, identify the map name in this command.
ipsec-pass-thru [*map_name*]	See IPsec Pass Through Inspection, page 13-30. If you added an IPsec Pass Through inspection policy map according to Configure an IPsec Pass Through Inspection Policy Map, page 13-31, identify the map name in this command.
ipv6 [*map_name*]	See IPv6 Inspection, page 13-33. If you added an IPv6 inspection policy map according to Configure an IPv6 Inspection Policy Map, page 13-34, identify the map name in this command.
mgcp [*map_name*]	See MGCP Inspection, page 14-12. If you added an MGCP inspection policy map according to Configuring an MGCP Inspection Policy Map for Additional Inspection Control, page 14-14, identify the map name in this command.
netbios [*map_name*]	See NetBIOS Inspection, page 13-37. If you added a NetBIOS inspection policy map according to Configure a NetBIOS Inspection Policy Map for Additional Inspection Control, page 13-37, identify the map name in this command.
pptp	See PPTP Inspection, page 13-39.
radius-accounting *map_name*	See RADIUS Accounting Inspection, page 15-13. The **radius-accounting** keyword is only available for a management class map. You must specify a RADIUS accounting inspection policy map; see Configure a RADIUS Accounting Inspection Policy Map, page 15-14.
rsh	See RSH Inspection, page 15-16.
rtsp [*map_name*]	See RTSP Inspection, page 14-17. If you added a RTSP inspection policy map according to Configure RTSP Inspection Policy Map, page 14-19, identify the map name in this command.

Table 12-2 Protocol Keywords

Keywords	Notes
scansafe [*map_name*] [**fail-open** | **fail-closed**]	If you want to enable ScanSafe (Cloud Web Security), use the procedure described in the following topic rather than this procedure: Configure a Service Policy to Send Traffic to Cloud Web Security, page 8-9. The cited procedure explains the full policy configuration, including how to configure the policy inspection map.
sip [*map_name*] [**tls-proxy** *proxy_name*]	See SIP Inspection, page 14-22. If you added a SIP inspection policy map according to Configure SIP Inspection Policy Map, page 14-24, identify the map name in this command. Specify a TLS proxy to enable inspection of encrypted traffic.
skinny [*map_name*] [**tls-proxy** *proxy_name*]	See Skinny (SCCP) Inspection, page 14-30. If you added a Skinny inspection policy map according to Configure a Skinny (SCCP) Inspection Policy Map for Additional Inspection Control, page 14-32, identify the map name in this command. Specify a TLS proxy to enable inspection of encrypted traffic.
snmp [*map_name*]	See SNMP Inspection, page 15-16. If you added an SNMP inspection policy map, identify the map name in this command.
sqlnet	See SQL*Net Inspection, page 15-18.
sunrpc	See Sun RPC Inspection, page 15-19. The default class map includes UDP port 111; if you want to enable Sun RPC inspection for TCP port 111, you need to create a new class map that matches TCP port 111, add the class to the policy, and then apply the **inspect sunrpc** command to that class.
tftp	See TFTP Inspection, page 13-45.
waas	Enables TCP option 33 parsing. Use when deploying Cisco Wide Area Application Services products.
xdmcp	See XDMCP Inspection, page 15-21.
vxlan	See VXLAN Inspection, page 15-22.

Note If you are editing the default global policy (or any in-use policy) to use a different inspection policy map, you must remove the old inspection with the **no inspect** *protocol* command, and then re-add it with the new inspection policy map name.

Step 6 To activate the policy map on one or more interfaces, enter the following command:

```
hostname(config)# service-policy policymap_name {global | interface interface_name}
```

Where **global** applies the policy map to all interfaces, and **interface** applies the policy to one interface. By default, the default policy map, "global_policy," is applied globally. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Choosing the Right Traffic Class for Inspection

The default Layer 3/4 class map for through traffic is called "inspection_default." It matches traffic using a special **match** command, **match default-inspection-traffic**, to match the default ports for each application protocol. This traffic class (along with **match any**, which is not typically used for inspection) matches both IPv4 and IPv6 traffic for inspections that support IPv6. See Guidelines for Application Inspection, page 12-5 for a list of IPv6-enabled inspections.

You can specify a **match access-list** command along with the **match default-inspection-traffic** command to narrow the matched traffic to specific IP addresses. Because the **match default-inspection-traffic** command specifies the ports to match, any ports in the ACL are ignored.

Tip We suggest that you only inspect traffic on ports on which you expect application traffic; if you inspect all traffic, for example using **match any**, the ASA performance can be impacted.

If you want to match non-standard ports, then create a new class map for the non-standard ports. See Default Inspections and NAT Limitations, page 12-6 for the standard ports for each inspection engine. You can combine multiple class maps in the same policy if desired, so you can create one class map to match certain traffic, and another to match different traffic. However, if traffic matches a class map that contains an inspection command, and then matches another class map that also has an inspection command, only the first matching class is used. For example, SNMP matches the inspection_default class. To enable SNMP inspection, enable SNMP inspection for the default class. Do not add another class that matches SNMP.

For example, to limit inspection to traffic from 10.1.1.0 to 192.168.1.0 using the default class map, enter the following commands:

```
hostname(config)# access-list inspect extended permit ip 10.1.1.0 255.255.255.0
192.168.1.0 255.255.255.0
hostname(config)# class-map inspection_default
hostname(config-cmap)# match access-list inspect
```

View the entire class map using the following command:

```
hostname(config-cmap)# show running-config class-map inspection_default
!
class-map inspection_default
 match default-inspection-traffic
 match access-list inspect
!
```

To inspect FTP traffic on port 21 as well as 1056 (a non-standard port), create an ACL that specifies the ports, and assign it to a new class map:

```
hostname(config)# access-list ftp_inspect extended permit tcp any any eq 21
hostname(config)# access-list ftp_inspect extended permit tcp any any eq 1056
hostname(config)# class-map new_inspection
hostname(config-cmap)# match access-list ftp_inspect
```

Configure Regular Expressions

Regular expressions define pattern matching for text strings. You can use these expressions in some protocol inspection maps to match packets based on strings such as URLs or the contents of particular header fields.

Create a Regular Expression

A regular expression matches text strings either literally as an exact string, or by using *metacharacters* so that you can match multiple variants of a text string. You can use a regular expression to match the content of certain application traffic; for example, you can match a URL string inside an HTTP packet.

Before You Begin

Use **Ctrl+V** to escape all of the special characters in the CLI, such as question mark (?) or a tab. For example, type **d[Ctrl+V]?g** to enter **d?g** in the configuration.

See the **regex** command in the command reference for performance impact information when matching a regular expression to packets. In general, matching against long input strings, or trying to match a large number of regular expressions, will reduce system performance.

Note As an optimization, the ASA searches on the deobfuscated URL. Deobfuscation compresses multiple forward slashes (/) into a single slash. For strings that commonly use double slashes, like “http://”, be sure to search for “http:/” instead.

The following table lists the metacharacters that have special meanings.

Table 12-3 Regular Expression Metacharacters

Character	Description	Notes
.	Dot	Matches any single character. For example, **d.g** matches dog, dag, dtg, and any word that contains those characters, such as doggonnit.
(*exp*)	Subexpression	A subexpression segregates characters from surrounding characters, so that you can use other metacharacters on the subexpression. For example, **d(o|a)g** matches dog and dag, but **do|ag** matches do and ag. A subexpression can also be used with repeat quantifiers to differentiate the characters meant for repetition. For example, **ab(xy){3}z** matches abxyxyxyz.
|	Alternation	Matches either expression it separates. For example, **dog|cat** matches dog or cat.
?	Question mark	A quantifier that indicates that there are 0 or 1 of the previous expression. For example, **lo?se** matches lse or lose.

Table 12-3 Regular Expression Metacharacters (continued)

Character	Description	Notes
*	Asterisk	A quantifier that indicates that there are 0, 1 or any number of the previous expression. For example, **lo*se** matches lse, lose, loose, and so on.
+	Plus	A quantifier that indicates that there is at least 1 of the previous expression. For example, **lo+se** matches lose and loose, but not lse.
{*x*} or {*x*,}	Minimum repeat quantifier	Repeat at least *x* times. For example, **ab(xy){2,}z** matches abxyxyz, abxyxyxyz, and so on.
[*abc*]	Character class	Matches any character in the brackets. For example, **[abc]** matches a, b, or c.
[^*abc*]	Negated character class	Matches a single character that is not contained within the brackets. For example, **[^abc]** matches any character other than a, b, or c. **[^A-Z]** matches any single character that is not an uppercase letter.
[*a*-*c*]	Character range class	Matches any character in the range. **[a-z]** matches any lowercase letter. You can mix characters and ranges: **[abcq-z]** matches a, b, c, q, r, s, t, u, v, w, x, y, z, and so does [**a-cq-z**]. The dash (-) character is literal only if it is the last or the first character within the brackets: **[abc-]** or **[-abc]**.
“”	Quotation marks	Preserves trailing or leading spaces in the string. For example, **“ test”** preserves the leading space when it looks for a match.
^	Caret	Specifies the beginning of a line.
\	Escape character	When used with a metacharacter, matches a literal character. For example, **\[** matches the left square bracket.
char	Character	When character is not a metacharacter, matches the literal character.
\r	Carriage return	Matches a carriage return 0x0d.
\n	Newline	Matches a new line 0x0a.
\t	Tab	Matches a tab 0x09.
\f	Formfeed	Matches a form feed 0x0c.
\x*NN*	Escaped hexadecimal number	Matches an ASCII character using hexadecimal (exactly two digits).
NNN	Escaped octal number	Matches an ASCII character as octal (exactly three digits). For example, the character 040 represents a space.

Procedure

Step 1 Test a regular expression to make sure it matches what you think it will match.

```
hostname(config)# test regex input_text regular_expression
```

Where the *input_text* argument is a string you want to match using the regular expression, up to 201 characters in length.

The *regular_expression* argument can be up to 100 characters in length.

Use **Ctrl+V** to escape all of the special characters in the CLI. For example, to enter a tab in the input text in the **test regex** command, you must enter **test regex "test[Ctrl+V Tab]" "test\t"**.

If the regular expression matches the input text, you see the following message:

```
INFO: Regular expression match succeeded.
```

If the regular expression does not match the input text, you see the following message:

```
INFO: Regular expression match failed.
```

Step 2 To add a regular expression after you tested it, enter the following command:

```
hostname(config)# regex name regular_expression
```

Where the *name* argument can be up to 40 characters in length.

The *regular_expression* argument can be up to 100 characters in length.

Examples

The following example creates two regular expressions for use in an inspection policy map:

```
hostname(config)# regex url_example example\.com
hostname(config)# regex url_example2 example2\.com
```

Create a Regular Expression Class Map

A regular expression class map identifies one or more regular expression. It is simply a collection of regular expression objects. You can use a regular expression class map in many cases in replace of a regular expression object.

Procedure

Step 1 Create the regular expression class map.

```
hostname(config)# class-map type regex match-any class_map_name
hostname(config-cmap)#
```

Where *class_map_name* is a string up to 40 characters in length. The name "class-default" is reserved. All types of class maps use the same name space, so you cannot reuse a name already used by another type of class map.

The **match-any** keyword specifies that the traffic matches the class map if it matches at least one of the regular expressions.

Step 2 (Optional) Add a description to the class map:

```
hostname(config-cmap)# description string
```

Step 3 Identify the regular expressions you want to include by entering the following command for each regular expression:

```
hostname(config-cmap)# match regex regex_name
```

Examples

The following example creates two regular expressions, and adds them to a regular expression class map. Traffic matches the class map if it includes the string “example.com” or “example2.com.”

```
hostname(config)# regex url_example example\.com
hostname(config)# regex url_example2 example2\.com
hostname(config)# class-map type regex match-any URLs
hostname(config-cmap)# match regex url_example
hostname(config-cmap)# match regex url_example2
```

History for Application Inspection

Feature Name	Releases	Description
Inspection policy maps	7.2(1)	The inspection policy map was introduced. The following command was introduced: **class-map type inspect**.
Regular expressions and policy maps	7.2(1)	Regular expressions and policy maps were introduced to be used under inspection policy maps. The following commands were introduced: **class-map type regex**, **regex**, **match regex**.
Match any for inspection policy maps	8.0(2)	The **match any** keyword was introduced for use with inspection policy maps: traffic can match one or more criteria to match the class map. Formerly, only **match all** was available.

Inspection of Basic Internet Protocols

The following topics explain application inspection for basic Internet protocols. For information on why you need to use inspection for certain protocols, and the overall methods for applying inspection, see Getting Started with Application Layer Protocol Inspection, page 12-1.

DNS Inspection

The following sections describe DNS application inspection.

DNS Inspection Actions

DNS inspection is enabled by default. You can customize DNS inspection to perform many tasks:

- Translate the DNS record based on the NAT configuration. For more information, see DNS and NAT, page 10-21.
- Enforce message length, domain-name length, and label length.
- Verify the integrity of the domain-name referred to by the pointer if compression pointers are encountered in the DNS message.
- Check to see if a compression pointer loop exists.
- Inspect packets based on the DNS header, type, class and more.

Defaults for DNS Inspection

DNS inspection is enabled by default, using the preset_dns_map inspection class map:

- The maximum DNS message length is 512 bytes.
- The maximum client DNS message length is automatically set to match the Resource Record.
- DNS Guard is enabled, so the ASA tears down the DNS session associated with a DNS query as soon as the DNS reply is forwarded by the ASA. The ASA also monitors the message exchange to ensure that the ID of the DNS reply matches the ID of the DNS query.
- Translation of the DNS record based on the NAT configuration is enabled.
- Protocol enforcement is enabled, which enables DNS message format check, including domain name length of no more than 255 characters, label length of 63 characters, compression, and looped pointer check.

See the following default DNS inspection commands:

```
class-map inspection_default
 match default-inspection-traffic
policy-map type inspect dns preset_dns_map
 parameters
    message-length maximum client auto
    message-length maximum 512
    dns-guard
    protocol-enforcement
    nat-rewrite
policy-map global_policy
 class inspection_default
  inspect dns preset_dns_map
! ...
service-policy global_policy global
```

Configure DNS Inspection

DNS inspection is enabled by default. You need to configure it only if you want non-default processing. If you want to customize DNS inspection, use the following process.

Procedure

Step 1 Configure DNS Inspection Policy Map, page 13-3.

Step 2 Configure the DNS Inspection Service Policy, page 13-6.

Configure DNS Inspection Policy Map

You can create a DNS inspection policy map to customize DNS inspection actions if the default inspection behavior is not sufficient for your network.

When defining traffic matching criteria, you can either create a class map or include the match statements directly in the policy map. The following procedure explains both approaches.

Before You Begin

Some traffic matching options use regular expressions for matching purposes. If you intend to use one of those techniques, first create the regular expression or regular expression class map.

Procedure

Step 1 (Optional) Create a DNS inspection class map by performing the following steps.

A class map groups multiple traffic matches.You can alternatively identify **match** commands directly in the policy map. The difference between creating a class map and defining the traffic match directly in the inspection policy map is that the class map lets you create more complex match criteria, and you can reuse class maps.

To specify traffic that should not match the class map, use the **match not** command. For example, if the **match not** command specifies the string “example.com,” then any traffic that includes “example.com” does not match the class map.

For the traffic that you identify in this class map, you specify actions to take on the traffic in the inspection policy map.

If you want to perform different actions for each **match** command, you should identify the traffic directly in the policy map.

a. Create the class map by entering the following command:

```
hostname(config)# class-map type inspect dns [match-all | match-any] class_map_name
hostname(config-cmap)#
```

Where *the class_map_name* is the name of the class map. The **match-all** keyword is the default, and specifies that traffic must match all criteria to match the class map. The match-any keyword specifies that the traffic matches the class map if it matches at least one **match** statement. The CLI enters class-map configuration mode, where you can enter one or more **match** commands.

b. (Optional) To add a description to the class map, enter the following command:

```
hostname(config-cmap)# description string
```

Where *string* is the description of the class map (up to 200 characters).

c. Specify the traffic on which you want to perform actions using one of the following **match** commands. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

- **match** [**not**] **header-flag** [**eq**] {*f_name* [*f_name*...] | *f_value*}—Matches the DNS flag. The *f_name* argument is the DNS flag name, one of the following: **AA** (Authoritative Answer), **QR** (Query), **RA** (Recursion Available), **RD** (Recursion Desired), **TC** (Truncation). The *f_value* argument is the 16-bit value in hex starting with 0x, from 0x0 to 0xffff. The **eq** keyword specifies an exact match (match all); without the **eq** keyword, the packet only needs to match one of the specified headers (match any). For example, **match header-flag AA QR**.
- **match** [**not**] **dns-type {eq** {*t_name* | *t_value*} | **range** *t_value1 t_value2*}—Matches the DNS type. The *t_name* argument is the DNS type name, one of the following: **A** (IPv4 address), **AXFR** (full zone transfer), **CNAME** (canonical name), **IXFR** (incremental zone transfer), **NS** (authoritative name server), **SOA** (start of a zone of authority) or **TSIG** (transaction signature). The *t_value* arguments are arbitrary values in the DNS type field (0-65535). The **range** keyword specifies a range, and the **eq** keyword specifies an exact match. For example: **match dns-type eq A**.
- **match** [**not**] **dns-class {eq** {**in** | *c_value*} | **range** *c_value1 c_value2*}—Matches the DNS class. The class is either **in** (for Internet) or *c_value,* an arbitrary value from 0 to 65535 in the DNS class field. The **range** keyword specifies a range, and the **eq** keyword specifies an exact match. For example: **match dns-class eq in**.
- **match** [**not**] {**question** | **resource-record** {**answer** | **authority** | **additional**}}—Matches a DNS question or resource record. The **question** keyword specifies the question portion of a DNS message. The **resource-record** keyword specifies one of these sections of the resource record: **answer**, **authority**, or **additional**. For example: **match resource-record answer**.
- **match** [**not**] **domain-name regex** {*regex_name* | **class** *class_name*}—Matches the DNS message domain name list against the specified regular expression or regular expression class.

d. Enter **exit** to leave class map configuration mode.

Step 2 Create a DNS inspection policy map, enter the following command:

```
hostname(config)# policy-map type inspect dns policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 3 (Optional) To add a description to the policy map, enter the following command:

```
hostname(config-pmap)# description string
```

Step 4 To apply actions to matching traffic, perform the following steps.

a. Specify the traffic on which you want to perform actions using one of the following methods:

- If you created a DNS class map, specify it by entering the following command:

```
hostname(config-pmap)# class class_map_name
hostname(config-pmap-c)#
```

- Specify traffic directly in the policy map using one of the **match** commands described for DNS class maps. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

b. Specify the action you want to perform on the matching traffic by entering the following command:

```
hostname(config-pmap-c)# {drop [log] | drop-connection [log] |
enforce-tsig {[drop] [log]} | mask [log] | log}
```

Not all options are available for each **match** or **class** command. See the CLI help or the command reference for the exact options available.

The **drop** keyword drops all packets that match.

The **drop-connection** keyword drops the packet and closes the connection.

The **mask** keyword masks out the matching portion of the packet. This action is available for header flag matches only.

The **log** keyword, which you can use alone or with one of the other keywords, sends a system log message.

The **enforce-tsig** {[**drop**] [**log**]} keyword enforces the presence of the TSIG resource record in a message. You can drop a packet without the TSIG resource record, log it, or drop and log it. You can use this option in conjunction with the mask action for header flag matches; otherwise, this action is exclusive with the other actions.

You can specify multiple **class** or **match** commands in the policy map. For information about the order of **class** and **match** commands, see How Multiple Traffic Classes are Handled, page 12-4.

For example:

```
hostname(config)# policy-map type inspect dns dns-map
hostname(config-pmap)# class dns-class-map
hostname(config-pmap-c)# drop
hostname(config-pmap-c)# match header-flag eq aa
hostname(config-pmap-c)# drop log
```

Step 5 To configure parameters that affect the inspection engine, perform the following steps:

a. To enter parameters configuration mode, enter the following command:

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b. Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- **dns-guard**—Enables DNS Guard. The ASA tears down the DNS session associated with a DNS query as soon as the DNS reply is forwarded by the ASA. The ASA also monitors the message exchange to ensure that the ID of the DNS reply matches the ID of the DNS query.
- **id-mismatch count** *number* **duration** *seconds* **action log**—Enables logging for excessive DNS ID mismatches, where the **count** *number* **duration** *seconds* arguments specify the maximum number of mismatch instances per second before a system message log is sent.
- **id-randomization**—Randomizes the DNS identifier for a DNS query.
- **message-length maximum** {*length* | **client** {*length* | **auto**} | **server** {*length* | **auto**}}—Sets the maximum DNS message length, from 512 to 65535 bytes. You can also set the maximum length for client or server messages. The **auto** keyword sets the maximum length to the value in the Resource Record.
- **nat-rewrite**—Translates the DNS record based on the NAT configuration.
- **protocol-enforcement**—Enables DNS message format check, including domain name length of no more than 255 characters, label length of 63 characters, compression, and looped pointer check.
- **tsig enforced action** {[**drop**] [**log**]}—Requires a TSIG resource record to be present. You can **drop** a non-conforming packet, **log** the packet, or both.

For example:

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)# dns-guard
hostname(config-pmap-p)# message-length maximum 1024
hostname(config-pmap-p)# nat-rewrite
hostname(config-pmap-p)# protocol-enforcement
```

Example

The following example shows a how to define a DNS inspection policy map.

```
regex domain_example "example\.com"
regex domain_foo "foo\.com"

! define the domain names that the server serves
class-map type inspect regex match-any my_domains
    match regex domain_example
    match regex domain_foo

! Define a DNS map for query only
class-map type inspect dns match-all pub_server_map
    match not header-flag QR
    match question
    match not domain-name regex class my_domains

policy-map type inspect dns new_dns_map
    class pub_server_map
        drop log
    match header-flag RD
        mask log
    parameters
        message-length maximum client auto
        message-length maximum 512
        dns-guard
        protocol-enforcement
        nat-rewrite
```

Configure the DNS Inspection Service Policy

The default ASA configuration includes DNS inspection on the default port applied globally on all interfaces. A common method for customizing the inspection configuration is to customize the default global policy. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

class-map *name*
match *parameter*

Example:

```
hostname(config)# class-map dns_class_map
hostname(config-cmap)# match access-list dns
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

policy-map *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for DNS inspection.

class *name*

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure DNS inspection.

inspect dns [*dns_policy_map*] [**dynamic-filter-snoop**]

Where:

- *dns_policy_map* is the optional DNS inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the DNS inspection policy map, see Configure DNS Inspection Policy Map, page 13-3.
- **dynamic-filter-snoop** enables dynamic filter snooping, which is used exclusively by the Botnet Traffic Filter. Include this keyword only if you use Botnet Traffic Filtering. We suggest that you enable DNS snooping only on interfaces where external DNS requests are going. Enabling DNS snooping on all UDP DNS traffic, including that going to an internal DNS server, creates unnecessary load on the ASA.

Example:

```
hostname(config-class)# no inspect dns
hostname(config-class)# inspect dns dns-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different DNS inspection policy map (for example, you are replacing the default preset_dns_map), you must remove the DNS inspection with the **no inspect dns** command, and then re-add it with the new DNS inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Examples

The following example shows a how to use a new inspection policy map in the global default configuration:

```
policy-map global_policy
 class inspection_default
  no inspect dns preset_dns_map
  inspect dns new_dns_map
service-policy global_policy global
```

Monitoring DNS Inspection

To view information about the current DNS connections, enter the following command:

```
hostname# show conn
```

For connections using a DNS server, the source port of the connection may be replaced by the IP address of the DNS server in the show conn command output.

A single connection is created for multiple DNS sessions, as long as they are between the same two hosts, and the sessions have the same 5-tuple (source/destination IP address, source/destination port, and protocol). DNS identification is tracked by app_id, and the idle timer for each app_id runs independently.

Because the app_id expires independently, a legitimate DNS response can only pass through the security appliance within a limited period of time and there is no resource build-up. However, when you enter the **show conn** command, you see the idle timer of a DNS connection being reset by a new DNS session. This is due to the nature of the shared DNS connection and is by design.

To display the statistics for DNS application inspection, enter the **show service-policy** command. The following is sample output from the **show service-policy** command:

```
hostname# show service-policy
Interface outside:
  Service-policy: sample_policy
    Class-map: dns_port
      Inspect: dns maximum-length 1500, packet 0, drop 0, reset-drop 0
```

FTP Inspection

The following sections describe the FTP inspection engine.

FTP Inspection Overview

The FTP application inspection inspects the FTP sessions and performs four tasks:

- Prepares dynamic secondary data connection

- Tracks the FTP command-response sequence
- Generates an audit trail
- Translates the embedded IP address

FTP application inspection prepares secondary channels for FTP data transfer. Ports for these channels are negotiated through PORT or PASV commands. The channels are allocated in response to a file upload, a file download, or a directory listing event.

Note If you disable FTP inspection engines with the **no inspect ftp** command, outbound users can start connections only in passive mode, and all inbound FTP is disabled.

Strict FTP

Strict FTP increases the security of protected networks by preventing web browsers from sending embedded commands in FTP requests. To enable strict FTP, include the **strict** option with the **inspect ftp** command.

When you use strict FTP, you can optionally specify an FTP inspection policy map to specify FTP commands that are not permitted to pass through the ASA.

After you enable the **strict** option on an interface, FTP inspection enforces the following behavior:

- An FTP command must be acknowledged before the ASA allows a new command.
- The ASA drops connections that send embedded commands.
- The 227 and PORT commands are checked to ensure they do not appear in an error string.

Caution Using the **strict** option may cause the failure of FTP clients that are not strictly compliant with FTP RFCs.

If the **strict** option is enabled, each FTP command and response sequence is tracked for the following anomalous activity:

- Truncated command—Number of commas in the PORT and PASV reply command is checked to see if it is five. If it is not five, then the PORT command is assumed to be truncated and the TCP connection is closed.
- Incorrect command—Checks the FTP command to see if it ends with <CR><LF> characters, as required by the RFC. If it does not, the connection is closed.
- Size of RETR and STOR commands—These are checked against a fixed constant. If the size is greater, then an error message is logged and the connection is closed.
- Command spoofing—The PORT command should always be sent from the client. The TCP connection is denied if a PORT command is sent from the server.
- Reply spoofing—PASV reply command (227) should always be sent from the server. The TCP connection is denied if a PASV reply command is sent from the client. This prevents the security hole when the user executes “227 xxxxx a1, a2, a3, a4, p1, p2.”
- TCP stream editing—The ASA closes the connection if it detects TCP stream editing.
- Invalid port negotiation—The negotiated dynamic port value is checked to see if it is less than 1024. As port numbers in the range from 1 to 1024 are reserved for well-known connections, if the negotiated port falls in this range, then the TCP connection is freed.

- Command pipelining—The number of characters present after the port numbers in the PORT and PASV reply command is cross checked with a constant value of 8. If it is more than 8, then the TCP connection is closed.
- The ASA replaces the FTP server response to the SYST command with a series of Xs. to prevent the server from revealing its system type to FTP clients. To override this default behavior, use the **no mask-syst-reply** command in the FTP map.

Configure FTP Inspection

FTP inspection is enabled by default. You need to configure it only if you want non-default processing. If you want to customize FTP inspection, use the following process.

Procedure

Step 1 Configure an FTP Inspection Policy Map, page 13-10.

Step 2 Configure the FTP Inspection Service Policy, page 13-13.

Configure an FTP Inspection Policy Map

FTP command filtering and security checks are provided using strict FTP inspection for improved security and control. Protocol conformance includes packet length checks, delimiters and packet format checks, command terminator checks, and command validation.

Blocking FTP based on user values is also supported so that it is possible for FTP sites to post files for download, but restrict access to certain users. You can block FTP connections based on file type, server name, and other attributes. System message logs are generated if an FTP connection is denied after inspection.

If you want FTP inspection to allow FTP servers to reveal their system type to FTP clients, and limit the allowed FTP commands, then create and configure an FTP inspection policy map. You can then apply the map when you enable FTP inspection.

Before You Begin

Some traffic matching options use regular expressions for matching purposes. If you intend to use one of those techniques, first create the regular expression or regular expression class map.

Procedure

Step 1 (Optional) Create an FTP inspection class map by performing the following steps.

A class map groups multiple traffic matches.You can alternatively identify **match** commands directly in the policy map. The difference between creating a class map and defining the traffic match directly in the inspection policy map is that the class map lets you create more complex match criteria, and you can reuse class maps.

To specify traffic that should not match the class map, use the **match not** command. For example, if the **match not** command specifies the string “example.com,” then any traffic that includes “example.com” does not match the class map.

For the traffic that you identify in this class map, you specify actions to take on the traffic in the inspection policy map.

If you want to perform different actions for each **match** command, you should identify the traffic directly in the policy map.

a. Create the class map by entering the following command:

```
hostname(config)# class-map type inspect ftp [match-all | match-any] class_map_name
hostname(config-cmap)#
```

Where *the class_map_name* is the name of the class map. The **match-all** keyword is the default, and specifies that traffic must match all criteria to match the class map. The match-any keyword specifies that the traffic matches the class map if it matches at least one **match** statement. The CLI enters class-map configuration mode, where you can enter one or more **match** commands.

b. (Optional) To add a description to the class map, enter the following command:

```
hostname(config-cmap)# description string
```

Where *string* is the description of the class map (up to 200 characters).

c. Specify the traffic on which you want to perform actions using one of the following **match** commands. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

- **match** [**not**] **filename regex** {*regex_name* | **class** *class_name*}—Matches the filename in the FTP transfer against the specified regular expression or regular expression class.
- **match** [**not**] **filetype regex** {*regex_name* | **class** *class_name*}—Matches the file type in the FTP transfer against the specified regular expression or regular expression class.
- **match** [**not**] **request-command** *ftp_command* [*ftp_command...*]—Matches the FTP command, one or more of the following:

  APPE—Append to a file.

  CDUP—Changes to the parent directory of the current working directory.

  DELE—Delete a file on the server.

  GET—Gets a file from the server.

  HELP—Provides help information.

  MKD—Makes a directory on the server.

  PUT—Sends a file to the server.

  RMD—Deletes a directory on the server.

  RNFR—Specifies the “rename-from” filename.

  RNTO—Specifies the “rename-to” filename.

  SITE—Used to specify a server-specific command. This is usually used for remote administration.

  STOU—Stores a file using a unique file name.
- **match** [**not**] **server regex** {*regex_name* | **class** *class_name*}—Matches the FTP server name against the specified regular expression or regular expression class.
- **match** [**not**] **username regex** {*regex_name* | **class** *class_name*}—Matches the FTP username against the specified regular expression or regular expression class.

d. Enter **exit** to leave class map configuration mode.

Step 2 Create an FTP inspection policy map:

```
hostname(config)# policy-map type inspect ftp policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 3 (Optional) To add a description to the policy map, enter the following command:

```
hostname(config-pmap)# description string
```

Step 4 To apply actions to matching traffic, perform the following steps.

a. Specify the traffic on which you want to perform actions using one of the following methods:

- If you created an FTP class map, specify it by entering the following command:

```
hostname(config-pmap)# class class_map_name
hostname(config-pmap-c)#
```

- Specify traffic directly in the policy map using one of the **match** commands described for FTP class maps. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

b. Specify the action you want to perform on the matching traffic by entering the following command:

```
hostname(config-pmap-c)# reset [log]
```

The **reset** keyword drops the packet, closes the connection, and sends a TCP reset to the server or client. Add the **log** keyword to send a system log message.

You can specify multiple **class** or **match** commands in the policy map. For information about the order of **class** and **match** commands, see How Multiple Traffic Classes are Handled, page 12-4.

Step 5 To configure parameters that affect the inspection engine, perform the following steps:

a. To enter parameters configuration mode, enter the following command:

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b. Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- **mask-banner**—Masks the greeting banner from the FTP server.
- **mask-syst-reply**—Masks the reply to **syst** command.

Example

Before submitting a username and password, all FTP users are presented with a greeting banner. By default, this banner includes version information useful to hackers trying to identify weaknesses in a system. The following example shows how to mask this banner:

```
hostname(config)# policy-map type inspect ftp mymap
hostname(config-pmap)# parameters
hostname(config-pmap-p)# mask-banner

hostname(config)# class-map match-all ftp-traffic
hostname(config-cmap)# match port tcp eq ftp

hostname(config)# policy-map ftp-policy
hostname(config-pmap)# class ftp-traffic
```

```
hostname(config-pmap-c)# inspect ftp strict mymap

hostname(config)# service-policy ftp-policy interface inside
```

Configure the FTP Inspection Service Policy

The default ASA configuration includes FTP inspection on the default port applied globally on all interfaces. A common method for customizing the inspection configuration is to customize the default global policy. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

class-map *name*
match *parameter*

Example:

```
hostname(config)# class-map ftp_class_map
hostname(config-cmap)# match access-list ftp
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

policy-map *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for FTP inspection.

class *name*

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure FTP inspection.

inspect ftp [**strict** [*ftp_policy_map*]]

Where:

- **strict** implements strict FTP. You must use strict FTP to specify an FTP inspection policy map.

- *ftp_policy_map* is the optional FTP inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the FTP inspection policy map, see Configure an FTP Inspection Policy Map, page 13-10.

Example:

```
hostname(config-class)# no inspect ftp
hostname(config-class)# inspect ftp strict ftp-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different FTP inspection policy map, you must remove the FTP inspection with the **no inspect ftp** command, and then re-add it with the new FTP inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Verifying and Monitoring FTP Inspection

FTP application inspection generates the following log messages:

- An Audit record 303002 is generated for each file that is retrieved or uploaded.
- The FTP command is checked to see if it is RETR or STOR and the retrieve and store commands are logged.
- The username is obtained by looking up a table providing the IP address.
- The username, source IP address, destination IP address, NAT address, and the file operation are logged.
- Audit record 201005 is generated if the secondary dynamic channel preparation failed due to memory shortage.

In conjunction with NAT, the FTP application inspection translates the IP address within the application payload. This is described in detail in RFC 959.

HTTP Inspection

The following sections describe the HTTP inspection engine.

HTTP Inspection Overview

Tip You can install a service module that performs application and URL filtering, which includes HTTP inspection, such as ASA CX or ASA FirePOWER. The HTTP inspection running on the ASA is not compatible with these modules. Note that it is far easier to configure application filtering using a purpose-built module rather than trying to manually configure it on the ASA using an HTTP inspection policy map.

Use the HTTP inspection engine to protect against specific attacks and other threats that are associated with HTTP traffic.

HTTP application inspection scans HTTP headers and body, and performs various checks on the data. These checks prevent various HTTP constructs, content types, and tunneling and messaging protocols from traversing the security appliance.

The enhanced HTTP inspection feature, which is also known as an application firewall and is available when you configure an HTTP inspection policy map, can help prevent attackers from using HTTP messages for circumventing network security policy.

HTTP application inspection can block tunneled applications and non-ASCII characters in HTTP requests and responses, preventing malicious content from reaching the web server. Size limiting of various elements in HTTP request and response headers, URL blocking, and HTTP server header type spoofing are also supported.

Enhanced HTTP inspection verifies the following for all HTTP messages:

- Conformance to RFC 2616
- Use of RFC-defined methods only.
- Compliance with the additional criteria.

Configure HTTP Inspection

HTTP inspection is not enabled by default. If you are not using a purpose-built module for HTTP inspection and application filtering, such as ASA CX or ASA FirePOWER, you can manually configure HTTP inspection on the ASA using the following process.

Tip Do not configure HTTP inspection in both a service module and on the ASA, as the inspections are not compatible.

Procedure

Step 1 Configure an HTTP Inspection Policy Map, page 13-16.

Step 2 Configure the HTTP Inspection Service Policy, page 13-19.

Configure an HTTP Inspection Policy Map

To specify actions when a message violates a parameter, create an HTTP inspection policy map. You can then apply the inspection policy map when you enable HTTP inspection.

Before You Begin

Some traffic matching options use regular expressions for matching purposes. If you intend to use one of those techniques, first create the regular expression or regular expression class map.

Procedure

Step 1 (Optional) Create an HTTP inspection class map by performing the following steps.

A class map groups multiple traffic matches.You can alternatively identify **match** commands directly in the policy map. The difference between creating a class map and defining the traffic match directly in the inspection policy map is that the class map lets you create more complex match criteria, and you can reuse class maps.

To specify traffic that should not match the class map, use the **match not** command. For example, if the **match not** command specifies the string "example.com," then any traffic that includes "example.com" does not match the class map.

For the traffic that you identify in this class map, you specify actions to take on the traffic in the inspection policy map.

If you want to perform different actions for each **match** command, you should identify the traffic directly in the policy map.

a. Create the class map by entering the following command:

```
hostname(config)# class-map type inspect http [match-all | match-any] class_map_name
hostname(config-cmap)#
```

Where the *class_map_name* is the name of the class map. The **match-all** keyword is the default, and specifies that traffic must match all criteria to match the class map. The match-any keyword specifies that the traffic matches the class map if it matches at least one **match** statement. The CLI enters class-map configuration mode, where you can enter one or more **match** commands.

b. (Optional) To add a description to the class map, enter the following command:

```
hostname(config-cmap)# description string
```

Where *string* is the description of the class map (up to 200 characters).

c. Specify the traffic on which you want to perform actions using one of the following **match** commands. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

- **match** [**not**] **req-resp content-type mismatch**—Matches traffic with a content-type field in the HTTP response that does not match the accept field in the corresponding HTTP request message.
- **match** [**not**] **request args regex** {*regex_name* | **class** *class_name*}—Matches text found in the HTTP request message arguments against the specified regular expression or regular expression class.
- **match** [**not**] **request body** {**regex** {*regex_name* | **class** *class_name*} | **length gt** *bytes*}—Matches text found in the HTTP request message body against the specified regular expression or regular expression class, or messages where the request body is greater than the specified length.

- **match** [**not**] **request header** {*field* | **regex** *regex_name*} **regex** {*regex_name* | **class** *class_name*}—Matches the content of a field in the HTTP request message header against the specified regular expression or regular expression class. You can specify the field name explicitly or match the field name to a regular expression. Field names are: accept, accept-charset, accept-encoding, accept-language, allow, authorization, cache-control, connection, content-encoding, content-language, content-length, content-location, content-md5, content-range, content-type, cookie, date, expect, expires, from, host, if-match, if-modified-since, if-none-match, if-range, if-unmodified-since, last-modified, max-forwards, pragma, proxy-authorization, range, referer, te, trailer, transfer-encoding, upgrade, user-agent, via, warning.
- **match** [**not**] **request header** {*field* | **regex** {*regex_name* | **class** *class_name*}} {**length gt** *bytes* | **count gt** *number*}—Matches the length of the specified fields in the HTTP request message header, or the overall number of fields (count) in the header. You can specify the field name explicitly or match the field name to a regular expression or regular expression class. Field names are listed in the previous bullet.
- **match** [**not**] **request header** {**length gt** *bytes* | **count gt** *number* | **non-ascii**}—Matches the overall length of the HTTP request message header, or the overall number of fields (count) in the header, or headers that have non-ASCII characters.
- **match** [**not**] **request method** {*method* | **regex** {*regex_name* | **class** *class_name*}}—Matches the HTTP request method. You can specify the method explicitly or match the method to a regular expression or regular expression class. Methods are: bcopy, bdelete, bmove, bpropfind, bproppatch, connect, copy, delete, edit, get, getattribute, getattributenames, getproperties, head, index, lock, mkcol, mkdir, move, notify, options, poll, post, propfind, proppatch, put, revadd, revlabel, revlog, revnum, save, search, setattribute, startrev, stoprev, subscribe, trace, unedit, unlock, unsubscribe.
- **match** [**not**] **request uri** {**regex** {*regex_name* | **class** *class_name*} | **length gt** *bytes*}—Matches text found in the HTTP request message URI against the specified regular expression or regular expression class, or messages where the request URI is greater than the specified length.
- **match** [**not**] **response body** {**active-x** | **java-applet** | **regex** {*regex_name* | **class** *class_name*}}—Matches text found in the HTTP response message body against the specified regular expression or regular expression class, or comments out Java applet and Active X object tags in order to filter them.
- **match** [**not**] **response body length gt** *bytes*—Matches HTTP response messages where the body is greater than the specified length.
- **match** [**not**] **response header** {*field* | **regex** *regex_name*} **regex** {*regex_name* | **class** *class_name*}—Matches the content of a field in the HTTP response message header against the specified regular expression or regular expression class. You can specify the field name explicitly or match the field name to a regular expression. Field names are: accept-ranges, age, allow, cache-control, connection, content-encoding, content-language, content-length, content-location, content-md5, content-range, content-type, date, etag, expires, last-modified, location, pragma, proxy-authenticate, retry-after, server, set-cookie, trailer, transfer-encoding, upgrade, vary, via, warning, www-authenticate.
- **match** [**not**] **response header** {*field* | **regex** {*regex_name* | **class** *class_name*}} {**length gt** *bytes* | **count gt** *number*}—Matches the length of the specified fields in the HTTP response message header, or the overall number of fields (count) in the header. You can specify the field name explicitly or match the field name to a regular expression or regular expression class. Field names are listed in the previous bullet.

- **match** [**not**] **response header** {**length gt** *bytes* | **count gt** *number* | **non-ascii**}—Matches the overall length of the HTTP response message header, or the overall number of fields (count) in the header, or headers that have non-ASCII characters.
- **match** [**not**] **response status-line regex** {*regex_name* | **class** *class_name*}—Matches text found in the HTTP response message status line against the specified regular expression or regular expression class.

d. Enter **exit** to leave class map configuration mode.

Step 2 Create an HTTP inspection policy map:

```
hostname(config)# policy-map type inspect http policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 3 (Optional) To add a description to the policy map, enter the following command:

```
hostname(config-pmap)# description string
```

Step 4 To apply actions to matching traffic, perform the following steps.

a. Specify the traffic on which you want to perform actions using one of the following methods:

- If you created an HTTP class map, specify it by entering the following command:

```
hostname(config-pmap)# class class_map_name
hostname(config-pmap-c)#
```

- Specify traffic directly in the policy map using one of the **match** commands described for HTTP class maps. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

b. Specify the action you want to perform on the matching traffic by entering the following command:

```
hostname(config-pmap-c)# {drop-connection [log]| reset [log] | log}
```

The **drop-connection** keyword drops the packet and closes the connection.

The **reset** keyword drops the packet, closes the connection, and sends a TCP reset to the server or client.

The **log** keyword, which you can use alone or with one of the other keywords, sends a system log message.

You can specify multiple **class** or **match** commands in the policy map. For information about the order of **class** and **match** commands, see How Multiple Traffic Classes are Handled, page 12-4.

Step 5 To configure parameters that affect the inspection engine, perform the following steps:

a. To enter parameters configuration mode, enter the following command:

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b. Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- **body-match-maximum** *number*—Sets the maximum number of characters in the body of an HTTP message that should be searched in a body match. The default is 200 bytes. A large number will have a significant impact on performance.

- **protocol-violation action** {**drop-connection** [**log**] | **reset** [**log**] | **log**}—Checks for HTTP protocol violations. You must also choose the action to take for violations (drop connection, reset, or log) and whether to enable or disable logging.
- **spoof-server** *string*—Substitutes a string for the server header field. WebVPN streams are not subject to the **spoof-server** command.

Example

The following example shows how to define an HTTP inspection policy map that will allow and log any HTTP connection that attempts to access “www\.xyz.com/.*\.asp" or "www\.xyz[0-9][0-9]\.com" with methods "GET" or "PUT." All other URL/Method combinations will be silently allowed.

```
hostname(config)# regex url1 “www\.xyz.com/.*\.asp”
hostname(config)# regex url2 “www\.xyz[0-9][0-9]\.com”
hostname(config)# regex get “GET”
hostname(config)# regex put “PUT”

hostname(config)# class-map type regex match-any url_to_log
hostname(config-cmap)# match regex url1
hostname(config-cmap)# match regex url2
hostname(config-cmap)# exit

hostname(config)# class-map type regex match-any methods_to_log
hostname(config-cmap)# match regex get
hostname(config-cmap)# match regex put
hostname(config-cmap)# exit

hostname(config)# class-map type inspect http http_url_policy
hostname(config-cmap)# match request uri regex class url_to_log
hostname(config-cmap)# match request method regex class methods_to_log
hostname(config-cmap)# exit

hostname(config)# policy-map type inspect http http_policy
hostname(config-pmap)# class http_url_policy
hostname(config-pmap-c)# log
```

Configure the HTTP Inspection Service Policy

HTTP inspection is not enabled in the default inspection policy, so you must enable it if you need this inspection. However, the default inspect class does include the default HTTP ports, so you can simply edit the default global inspection policy to add HTTP inspection. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

```
class-map name
match parameter
```

Example:

```
hostname(config)# class-map http_class_map
hostname(config-cmap)# match access-list http
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

policy-map *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for HTTP inspection.

class *name*

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure HTTP inspection.

inspect http [*http_policy_map*]

Where *http_policy_map* is the optional HTTP inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the HTTP inspection policy map, see Configure an HTTP Inspection Policy Map, page 13-16.

Example:

```
hostname(config-class)# no inspect http
hostname(config-class)# inspect http http-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different HTTP inspection policy map, you must remove the HTTP inspection with the **no inspect http** command, and then re-add it with the new HTTP inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

ICMP Inspection

The ICMP inspection engine allows ICMP traffic to have a "session" so it can be inspected like TCP and UDP traffic. Without the ICMP inspection engine, we recommend that you do not allow ICMP through the ASA in an ACL. Without stateful inspection, ICMP can be used to attack your network. The ICMP inspection engine ensures that there is only one response for each request, and that the sequence number is correct.

However, ICMP traffic directed to an ASA interface is never inspected, even if you enable ICMP inspection. Thus, a ping (echo request) to an interface can fail under specific circumstances, such as when the echo request comes from a source that the ASA can reach through a backup default route.

For information on enabling ICMP inspection, see Configure Application Layer Protocol Inspection, page 12-9.

ICMP Error Inspection

When ICMP Error inspection is enabled, the ASA creates translation sessions for intermediate hops that send ICMP error messages, based on the NAT configuration. The ASA overwrites the packet with the translated IP addresses.

When disabled, the ASA does not create translation sessions for intermediate nodes that generate ICMP error messages. ICMP error messages generated by the intermediate nodes between the inside host and the ASA reach the outside host without consuming any additional NAT resource. This is undesirable when an outside host uses the traceroute command to trace the hops to the destination on the inside of the ASA. When the ASA does not translate the intermediate hops, all the intermediate hops appear with the mapped destination IP address.

The ICMP payload is scanned to retrieve the five-tuple from the original packet. Using the retrieved five-tuple, a lookup is performed to determine the original address of the client. The ICMP error inspection engine makes the following changes to the ICMP packet:

- In the IP Header, the mapped IP is changed to the real IP (Destination Address) and the IP checksum is modified.
- In the ICMP Header, the ICMP checksum is modified due to the changes in the ICMP packet.
- In the Payload, the following changes are made:
  - Original packet mapped IP is changed to the real IP
  - Original packet mapped port is changed to the real Port
  - Original packet IP checksum is recalculated

For information on enabling ICMP Error inspection, see Configure Application Layer Protocol Inspection, page 12-9.

Instant Messaging Inspection

The Instant Messaging (IM) inspect engine lets you control the network usage of IM and stop leakage of confidential data, propagation of worms, and other threats to the corporate network.

IM inspection is not enabled by default. You must configure it if you want IM inspection.

Procedure

Step 1 Configure an Instant Messaging Inspection Policy Map, page 13-22.

Step 2 Configure the IM Inspection Service Policy, page 13-24.

Configure an Instant Messaging Inspection Policy Map

To specify actions when a message violates a parameter, create an IM inspection policy map. You can then apply the inspection policy map when you enable IM inspection.

Before You Begin

Some traffic matching options use regular expressions for matching purposes. If you intend to use one of those techniques, first create the regular expression or regular expression class map.

Procedure

Step 1 (Optional) Create an IM inspection class map by performing the following steps.

A class map groups multiple traffic matches.You can alternatively identify **match** commands directly in the policy map. The difference between creating a class map and defining the traffic match directly in the inspection policy map is that the class map lets you create more complex match criteria, and you can reuse class maps.

To specify traffic that should not match the class map, use the **match not** command. For example, if the **match not** command specifies the string “example.com,” then any traffic that includes “example.com” does not match the class map.

For the traffic that you identify in this class map, you specify actions to take on the traffic in the inspection policy map.

If you want to perform different actions for each **match** command, you should identify the traffic directly in the policy map.

a. Create the class map by entering the following command:

```
hostname(config)# class-map type inspect im [match-all | match-any] class_map_name
hostname(config-cmap)#
```

Where *the class_map_name* is the name of the class map. The **match-all** keyword is the default, and specifies that traffic must match all criteria to match the class map. The match-any keyword specifies that the traffic matches the class map if it matches at least one **match** statement. The CLI enters class-map configuration mode, where you can enter one or more **match** commands.

b. (Optional) To add a description to the class map, enter the following command:

```
hostname(config-cmap)# description string
```

Where *string* is the description of the class map (up to 200 characters).

c. Specify the traffic on which you want to perform actions using one of the following **match** commands. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

- **match** [**not**] **protocol {im-yahoo | im-msn}**—Matches a specific IM protocol, either Yahoo or MSN.

- **match** [**not**] **service {chat | file-transfer | webcam | voice-chat | conference | games}**—Matches the specific IM service.
- **match** [**not**] **login-name regex** {*regex_name* | **class** *class_name*}—Matches the source client login name of the IM message against the specified regular expression or regular expression class.
- **match** [**not**] **peer-login-name regex** {*regex_name* | **class** *class_name*}—Matches the destination peer login name of the IM message against the specified regular expression or regular expression class.
- **match** [**not**] **ip-address** *ip_address mask*}—Matches the source IP address and mask of the IM message.
- **match** [**not**] **peer-ip-address** *ip_address mask*}—Matches the destination IP address and mask of the IM message.
- **match** [**not**] **version regex** {*regex_name* | **class** *class_name*}—Matches the version of the IM message against the specified regular expression or regular expression class.
- **match** [**not**] **filename regex** {*regex_name* | **class** *class_name*}—Matches the filename of the IM message against the specified regular expression or regular expression class. This match is not supported for the MSN IM protocol.

d. Enter **exit** to leave class map configuration mode.

Step 2 Create an IM inspection policy map:

```
hostname(config)# policy-map type inspect im policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 3 (Optional) To add a description to the policy map, enter the following command:

```
hostname(config-pmap)# description string
```

Step 4 To apply actions to matching traffic, perform the following steps.

a. Specify the traffic on which you want to perform actions using one of the following methods:

- If you created an IM class map, specify it by entering the following command:

```
hostname(config-pmap)# class class_map_name
hostname(config-pmap-c)#
```

- Specify traffic directly in the policy map using one of the **match** commands described for IM class maps. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

b. Specify the action you want to perform on the matching traffic by entering the following command:

```
hostname(config-pmap-c)# {drop-connection [log]| reset [log] | log}
```

The **drop-connection** keyword drops the packet and closes the connection.

The **reset** keyword drops the packet, closes the connection, and sends a TCP reset to the server or client.

The **log** keyword, which you can use alone or with one of the other keywords, sends a system log message.

You can specify multiple **class** or **match** commands in the policy map. For information about the order of **class** and **match** commands, see How Multiple Traffic Classes are Handled, page 12-4.

Example

The following example shows how to define an IM inspection policy map.

```
hostname(config)# regex loginname1 "ying\@yahoo.com"
hostname(config)# regex loginname2 "Kevin\@yahoo.com"
hostname(config)# regex loginname3 "rahul\@yahoo.com"
hostname(config)# regex loginname4 "darshant\@yahoo.com"
hostname(config)# regex yahoo_version_regex "1\.0"
hostname(config)# regex gif_files ".*\.gif"
hostname(config)# regex exe_files ".*\.exe"

hostname(config)# class-map type regex match-any yahoo_src_login_name_regex
hostname(config-cmap)# match regex loginname1
hostname(config-cmap)# match regex loginname2

hostname(config)# class-map type regex match-any yahoo_dst_login_name_regex
hostname(config-cmap)# match regex loginname3
hostname(config-cmap)# match regex loginname4

hostname(config)# class-map type inspect im match-any yahoo_file_block_list
hostname(config-cmap)# match filename regex gif_files
hostname(config-cmap)# match filename regex exe_files

hostname(config)# class-map type inspect im match-all yahoo_im_policy
hostname(config-cmap)# match login-name regex class yahoo_src_login_name_regex
hostname(config-cmap)# match peer-login-name regex class yahoo_dst_login_name_regex

hostname(config)# class-map type inspect im match-all yahoo_im_policy2
hostname(config-cmap)# match version regex yahoo_version_regex

hostname(config)# class-map im_inspect_class_map
hostname(config-cmap)# match default-inspection-traffic

hostname(config)# policy-map type inspect im im_policy_all
hostname(config-pmap)# class yahoo_file_block_list
hostname(config-pmap-c)# match service file-transfer
hostname(config-pmap)# class yahoo_im_policy
hostname(config-pmap-c)# drop-connection
hostname(config-pmap)# class yahoo_im_policy2
hostname(config-pmap-c)# reset
hostname(config)# policy-map global_policy_name
hostname(config-pmap)# class im_inspect_class_map
hostname(config-pmap-c)# inspect im im_policy_all
```

Configure the IM Inspection Service Policy

IM inspection is not enabled in the default inspection policy, so you must enable it if you need this inspection. However, the default inspect class does include the default IM ports, so you can simply edit the default global inspection policy to add IM inspection. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

```
class-map name
match parameter
```

Example:

```
hostname(config)# class-map im_class_map
hostname(config-cmap)# match access-list im
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

```
policy-map name
```

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for IM inspection.

```
class name
```

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure IM inspection.

```
inspect im [im_policy_map]
```

Where *im_policy_map* is the optional IM inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the IM inspection policy map, see Configure an Instant Messaging Inspection Policy Map, page 13-22.

Example:

```
hostname(config-class)# no inspect im
hostname(config-class)# inspect im im-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different IM inspection policy map, you must remove the IM inspection with the **no inspect im** command, and then re-add it with the new IM inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

```
service-policy policymap_name {global | interface interface_name}
```

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

IP Options Inspection

You can configure IP Options inspection to control which IP packets with specific IP options are allowed through the ASA. Configuring this inspection instructs the ASA to allow a packet to pass or to clear the specified IP options and then allow the packet to pass.

The following sections describe the IP Options inspection engine.

IP Options Inspection Overview

Each IP packet contains an IP header with the Options field. The Options field, commonly referred to as IP Options, provide for control functions that are required in some situations but unnecessary for most common communications. In particular, IP Options include provisions for time stamps, security, and special routing. Use of IP Options is optional, and the field can contain zero, one, or more options.

For a list of IP options, with references to the relevant RFCs, see the IANA page, http://www.iana.org/assignments/ip-parameters/ip-parameters.xhtml.

You can configure IP Options inspection to control which IP packets with specific IP options are allowed through the ASA. Configuring this inspection instructs the ASA to allow a packet to pass or to clear the specified IP options and then allow the packet to pass.

What Happens When You Clear an Option

When you configure an IP options inspection policy map, you can specify whether you want to allow or clear each option type. If you do not specify an option type, packets that contain the option are dropped.

If you simply allow an option, packets containing the option are passed through unchanged.

If you specify that you want to clear an option from IP headers, the IP header changes in the following ways:

- The option is removed from the header.
- The Options field is padded so that the field ends on a 32 bit boundary.
- Internet header length (IHL) in the packet changes.
- The total length of the packet changes.

- The checksum is recomputed.

Supported IP Options for Inspection

IP Options inspection can check for the following IP options in a packet. If an IP header contains additional options other than these, regardless of whether the ASA is configured to allow these options, the ASA will drop the packet.

- End of Options List (EOOL) or IP Option 0—This option, which contains just a single zero byte, appears at the end of all options to mark the end of a list of options. This might not coincide with the end of the header according to the header length.
- No Operation (NOP) or IP Option 1—The Options field in the IP header can contain zero, one, or more options, which makes the total length of the field variable. However, the IP header must be a multiple of 32 bits. If the number of bits of all options is not a multiple of 32 bits, the NOP option is used as "internal padding" to align the options on a 32-bit boundary.
- Router Alert (RTRALT) or IP Option 20—This option notifies transit routers to inspect the contents of the packet even when the packet is not destined for that router. This inspection is valuable when implementing RSVP and similar protocols that require relatively complex processing from the routers along the packet's delivery path. Dropping RSVP packets containing the Router Alert option can cause problems in VoIP implementations.

Defaults for IP Options Inspection

IP Options inspection is enabled by default, using the _default_ip_options_map inspection policy map.

- The Router Alert option is allowed.
- Packets that contain any other options are dropped. This includes packets that contain unsupported options.

Following is the policy map configuration:

```
policy-map type inspect ip-options _default_ip_options_map
 description Default IP-OPTIONS policy-map
 parameters
  router-alert action allow
```

Configure IP Options Inspection

IP options inspection is enabled by default. You need to configure it only if you want to allow additional options than the default map allows.

Procedure

Step 1 Configure an IP Options Inspection Policy Map, page 13-28.

Step 2 Configure the IP Options Inspection Service Policy, page 13-28.

Configure an IP Options Inspection Policy Map

If you want to perform non-default IP options inspection, create an IP options inspection policy map to specify how you want to handle each supported option type.

Procedure

Step 1 Create an IP options inspection policy map:

```
hostname(config)# policy-map type inspect ip-options policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 2 (Optional) To add a description to the policy map, enter the following command:

```
hostname(config-pmap)# description string
```

Step 3 To configure parameters that affect the inspection engine, perform the following steps:

a. To enter parameters configuration mode, enter the following command:

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b. Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option. In all cases, the **allow** action allows packets that contain the option without modification; the **clear** action allows the packets but removes the option from the header. Any packet that contains an option that you do not include in the map is dropped. For a description of the options, see Supported IP Options for Inspection, page 13-27.

- **eool action** {**allow** | **clear**}—Allows or clears the End of Options List option.
- **nop action** {**allow** | **clear**}—Allows or clears the No Operation option.
- **router-alert action** {**allow** | **clear**}—Allows or clears the Router Alert (RTRALT) option.

Configure the IP Options Inspection Service Policy

The default ASA configuration includes IP options inspection applied globally on all interfaces. A common method for customizing the inspection configuration is to customize the default global policy. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

```
class-map name
match parameter
```

Example:

```
hostname(config)# class-map ip_options_class_map
hostname(config-cmap)# match access-list ipoptions
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

policy-map *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for IP options inspection.

class *name*

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure IP options inspection.

inspect ip-options [*ip_options_policy_map*]

Where *ip_options_policy_map* is the optional IP options inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the IP options inspection policy map, see Configure an IP Options Inspection Policy Map, page 13-28.

Example:

```
hostname(config-class)# no inspect ip-options
hostname(config-class)# inspect ip-options ip-options-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different IP options inspection policy map, you must remove the IP options inspection with the **no inspect ip-options** command, and then re-add it with the new IP options inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Monitoring IP Options Inspection

You can use these techniques to monitor the results of IP options inspection:

- Each time a packet is dropped due to inspection, syslog 106012 is issued. The message shows which option caused the drop.
- Use the **show service-policy inspect ip-options** command to view statistics for each option.

IPsec Pass Through Inspection

The following sections describe the IPsec Pass Through inspection engine.

- IPsec Pass Through Inspection Overview, page 13-30
- Configure IPsec Pass Through Inspection, page 13-30

IPsec Pass Through Inspection Overview

Internet Protocol Security (IPsec) is a protocol suite for securing IP communications by authenticating and encrypting each IP packet of a data stream. IPsec also includes protocols for establishing mutual authentication between agents at the beginning of the session and negotiation of cryptographic keys to be used during the session. IPsec can be used to protect data flows between a pair of hosts (for example, computer users or servers), between a pair of security gateways (such as routers or firewalls), or between a security gateway and a host.

IPsec Pass Through application inspection provides convenient traversal of ESP (IP protocol 50) and AH (IP protocol 51) traffic associated with an IKE UDP port 500 connection. It avoids lengthy ACL configuration to permit ESP and AH traffic and also provides security using timeout and max connections.

Configure a policy map for IPsec Pass Through to specify the restrictions for ESP or AH traffic. You can set the per client max connections and the idle timeout.

NAT and non-NAT traffic is permitted. However, PAT is not supported.

Configure IPsec Pass Through Inspection

IPsec Pass Through inspection is not enabled by default. You must configure it if you want IPsec Pass Through inspection.

Procedure

Step 1 Configure an IPsec Pass Through Inspection Policy Map, page 13-31.

Step 2 Configure the IPsec Pass Through Inspection Service Policy, page 13-32.

Configure an IPsec Pass Through Inspection Policy Map

An IPsec Pass Through map lets you change the default configuration values used for IPsec Pass Through application inspection. You can use an IPsec Pass Through map to permit certain flows without using an ACL.

The configuration includes a default map, _default_ipsec_passthru_map, that sets no maximum limit on ESP connections per client, and sets the ESP idle timeout at 10 minutes. You need to configure an inspection policy map only if you want different values, or if you want to set AH values.

Procedure

Step 1 Create an IPsec Pass Through inspection policy map:

```
hostname(config)# policy-map type inspect ipsec-pass-thru policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 2 (Optional) To add a description to the policy map, enter the following command:

```
hostname(config-pmap)# description string
```

Step 3 To configure parameters that affect the inspection engine, perform the following steps:

a. To enter parameters configuration mode, enter the following command:

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b. Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- **esp per-client-max** *number* **timeout** *time*—Allows ESP tunnels and sets the maximum connections allowed per client and the idle timeout (in hh:mm:ss format). To allow an unlimited number of connections, specify 0 for the number.
- **ah per-client-max** *number* **timeout** *time*—Allows AH tunnels. The parameters have the same meaning as for the esp command.

Example

The following example shows how to use ACLs to identify IKE traffic, define an IPsec Pass Thru parameter map, define a policy, and apply the policy to the outside interface:

```
hostname(config)# access-list ipsecpassthruacl permit udp any any eq 500
hostname(config)# class-map ipsecpassthru-traffic
hostname(config-cmap)# match access-list ipsecpassthruacl
hostname(config)# policy-map type inspect ipsec-pass-thru iptmap
hostname(config-pmap)# parameters
hostname(config-pmap-p)# esp per-client-max 10 timeout 0:11:00
hostname(config-pmap-p)# ah per-client-max 5 timeout 0:06:00
hostname(config)# policy-map inspection_policy
hostname(config-pmap)# class ipsecpassthru-traffic
hostname(config-pmap-c)# inspect ipsec-pass-thru iptmap
hostname(config)# service-policy inspection_policy interface outside
```

Configure the IPsec Pass Through Inspection Service Policy

IPsec Pass Through inspection is not enabled in the default inspection policy, so you must enable it if you need this inspection. However, the default inspect class does include the default IPsec ports, so you can simply edit the default global inspection policy to add IPsec inspection. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

```
class-map name
match parameter
```

Example:

```
hostname(config)# class-map ipsec_class_map
hostname(config-cmap)# match access-list ipsec
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

```
policy-map name
```

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for IPsec Pass Through inspection.

```
class name
```

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure IPsec Pass Through inspection.

```
inspect ipsec-pass-thru [ipsec_policy_map]
```

Where *ipsec_policy_map* is the optional IPsec Pass Through inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the inspection policy map, see Configure an IPsec Pass Through Inspection Policy Map, page 13-31.

Example:

```
hostname(config-class)# no inspect ipsec-pass-thru
hostname(config-class)# inspect ipsec-pass-thru ipsec-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different IPsec Pass Through inspection policy map, you must remove the IPsec Pass Through inspection with the **no inspect ipsec-pass-thru** command, and then re-add it with the new IPsec Pass Thruough inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

IPv6 Inspection

IPv6 inspection lets you selectively log or drop IPv6 traffic based on the extension header. In addition, IPv6 inspection can check conformance to RFC 2460 for type and order of extension headers in IPv6 packets.

- Defaults for IPv6 Inspection, page 13-33
- Configure IPv6 Inspection, page 13-34

Defaults for IPv6 Inspection

If you enable IPv6 inspection and do not specify an inspection policy map, then the default IPv6 inspection policy map is used, and the following actions are taken:

- Allows only known IPv6 extension headers. Non-conforming packets are dropped and logged.
- Enforces the order of IPv6 extension headers as defined in the RFC 2460 specification. Non-conforming packets are dropped and logged.
- Drops any packet with a routing type header.

Following is the policy map configuration:

```
policy-map type inspect ipv6 _default_ipv6_map
 description Default IPV6 policy-map
 parameters
  verify-header type
  verify-header order
 match header routing-type range 0 255
  drop log
```

Configure IPv6 Inspection

IPv6 inspection is not enabled by default. You must configure it if you want IPv6 inspection.

Procedure

Step 1 Configure an IPv6 Inspection Policy Map, page 13-34.

Step 2 Configure the IPv6 Inspection Service Policy, page 13-35.

Configure an IPv6 Inspection Policy Map

To identify extension headers to drop or log, or to disable packet verification, create an IPv6 inspection policy map to be used by the service policy.

Procedure

Step 1 Create an IPv6 inspection policy map.

```
hostname(config)# policy-map type inspect ipv6 policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 2 (Optional) Add a description to the policy map.

```
hostname(config-pmap)# description string
```

Step 3 (Optional) Drop or log traffic based on the headers in IPv6 messages.

a. Identify the traffic based on the IPv6 header.

```
hostname(config-pmap)# match header type
```

Where type is one of the following:

- **ah**—Matches the IPv6 Authentication extension header.
- **count gt** *number*—Specifies the maximum number of IPv6 extension headers, from 0 to 255.
- **destination-option**—Matches the IPv6 destination-option extension header.
- **esp**—Matches the IPv6 Encapsulation Security Payload (ESP) extension header.
- **fragment**—Matches the IPv6 fragment extension header.
- **hop-by-hop**—Matches the IPv6 hop-by-hop extension header.
- **routing-address count gt** *number*—Sets the maximum number of IPv6 routing header type 0 addresses, greater than a number between 0 and 255.
- **routing-type** {**eq** | **range**} *number*—Matches the IPv6 routing header type, from 0 to 255. For a range, separate values by a space, for example, **30 40**.

b. Specify the action to perform on matching packets. You can drop the packet and optionally log it, or just log it. If you do not enter an action, the packet is logged.

```
hostname(config-pmap)# {drop [log] | log}
```

c. Repeat the process until you identify all headers that you want to drop or log.

Step 4 Configure parameters that affect the inspection engine.

a. Enter parameters configuration mode.

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b. Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- **verify-header type**—Allows only known IPv6 extension headers.
- **verify-header order**—Enforces the order of IPv6 extension headers as defined in RFC 2460.

Examples

The following example creates an inspection policy map that will drop and log all IPv6 packets with the hop-by-hop, destination-option, routing-address, and routing type 0 headers. It also enforces header order and type.

```
policy-map type inspect ipv6 ipv6-pm
 parameters
  verify-header type
  verify-header order
 match header hop-by-hop
  drop log
 match header destination-option
  drop log
 match header routing-address count gt 0
  drop log
 match header routing-type eq 0
  drop log

policy-map global_policy
 class class-default
  inspect ipv6 ipv6-pm
!
service-policy global_policy global
```

Configure the IPv6 Inspection Service Policy

IPv6 inspection is not enabled in the default inspection policy, so you must enable it if you need this inspection. You can simply edit the default global inspection policy to add IPv6 inspection. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

class-map *name*
match *parameter*

Example:

```
hostname(config)# class-map ipv6_class_map
hostname(config-cmap)# match access-list ipv6
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

policy-map *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for IPv6 inspection.

class *name*

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure IPv6 inspection.

inspect ipv6 [*ipv6_policy_map*]

Where *ipv6_policy_map* is the optional IPv6 inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the inspection policy map, see Configure an IPv6 Inspection Policy Map, page 13-34.

Example:

```
hostname(config-class)# no inspect ipv6
hostname(config-class)# inspect ipv6 ipv6-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different IPv6 inspection policy map, you must remove the IPv6 inspection with the **no inspect ipv6** command, and then re-add it with the new IPv6 inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

NetBIOS Inspection

NetBIOS inspection is enabled by default. The NetBIOS inspection engine translates IP addresses in the NetBIOS name service (NBNS) packets according to the ASA NAT configuration. You can optionally create a policy map to drop or log NetBIOS protocol violations.

Procedure

Step 1 Configure a NetBIOS Inspection Policy Map for Additional Inspection Control, page 13-37.

Step 2 Configure the NetBIOS Inspection Service Policy, page 13-38.

Configure a NetBIOS Inspection Policy Map for Additional Inspection Control

To specify the action for protocol violations, create a NETBIOS inspection policy map. You can then apply the inspection policy map when you enable NETBIOS inspection.

Procedure

Step 1 Create a NetBIOS inspection policy map.

```
hostname(config)# policy-map type inspect netbios policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 2 (Optional) To add a description to the policy map, enter the following command:

```
hostname(config-pmap)# description string
```

Step 3 Enter parameters configuration mode.

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

Step 4 Specify the action to take for NETBIOS protocol violations.

```
hostname(config-pmap-p)# protocol-violation action {drop [log] | log}
```

Where the **drop** action drops the packet. The **log** action sends a system log message when this policy map matches traffic.

Example

```
hostname(config)# policy-map type inspect netbios netbios_map
hostname(config-pmap)# parameters
hostname(config-pmap-p)# protocol-violation drop log

hostname(config)# policy-map netbios_policy
hostname(config-pmap)# class inspection_default
hostname(config-pmap-c)# inspect netbios netbios_map
```

Configure the NetBIOS Inspection Service Policy

NetBIOS application inspection performs NAT for the embedded IP address in the NetBIOS name service packets and NetBIOS datagram services packets. It also enforces protocol conformance, checking the various count and length fields for consistency.

The default ASA configuration includes NetBIOS inspection on the default port applied globally on all interfaces. A common method for customizing the inspection configuration is to customize the default global policy. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

```
class-map name
match parameter
```

Example:

```
hostname(config)# class-map netbios_class_map
hostname(config-cmap)# match access-list netbios
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

```
policy-map name
```

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for NetBIOS inspection.

```
class name
```

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure NetBIOS inspection.

```
inspect netbios [netbios_policy_map]
```

Where *netbios_policy_map* is the optional NetBIOS inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the NetBIOS inspection policy map, see Configure a NetBIOS Inspection Policy Map for Additional Inspection Control, page 13-37.

Example:

```
hostname(config-class)# no inspect netbios
```

```
hostname(config-class)# inspect netbios netbios-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different NetBIOS inspection policy map, you must remove the NetBIOS inspection with the **no inspect skinny** command, and then re-add it with the new NetBIOS inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

PPTP Inspection

PPTP is a protocol for tunneling PPP traffic. A PPTP session is composed of one TCP channel and usually two PPTP GRE tunnels. The TCP channel is the control channel used for negotiating and managing the PPTP GRE tunnels. The GRE tunnels carry PPP sessions between the two hosts.

When enabled, PPTP application inspection inspects PPTP protocol packets and dynamically creates the GRE connections and xlates necessary to permit PPTP traffic.

Specifically, the ASA inspects the PPTP version announcements and the outgoing call request/response sequence. Only PPTP Version 1, as defined in RFC 2637, is inspected. Further inspection on the TCP control channel is disabled if the version announced by either side is not Version 1. In addition, the outgoing-call request and reply sequence are tracked. Connections and xlates are dynamically allocated as necessary to permit subsequent secondary GRE data traffic.

The PPTP inspection engine must be enabled for PPTP traffic to be translated by PAT. Additionally, PAT is only performed for a modified version of GRE (RFC2637) and only if it is negotiated over the PPTP TCP control channel. PAT is not performed for the unmodified version of GRE (RFC 1701 and RFC 1702).

For information on enabling PPTP inspection, see Configure Application Layer Protocol Inspection, page 12-9.

SMTP and Extended SMTP Inspection

ESMTP inspection detects attacks, including spam, phising, malformed message attacks, buffer overflow/underflow attacks. It also provides support for application security and protocol conformance, which enforce the sanity of the ESMTP messages as well as detect several attacks, block senders/receivers, and block mail relay.

The following sections describe the ESMTP inspection engine.

- SMTP and ESMTP Inspection Overview, page 13-40

SMTP and ESMTP Inspection Overview

ESMTP application inspection provides improved protection against SMTP-based attacks by restricting the types of SMTP commands that can pass through the ASA and by adding monitoring capabilities.

ESMTP is an enhancement to the SMTP protocol and is similar is most respects to SMTP. For convenience, the term SMTP is used in this document to refer to both SMTP and ESMTP. The application inspection process for extended SMTP is similar to SMTP application inspection and includes support for SMTP sessions. Most commands used in an extended SMTP session are the same as those used in an SMTP session but an ESMTP session is considerably faster and offers more options related to reliability and security, such as delivery status notification.

Extended SMTP application inspection adds support for these extended SMTP commands, including AUTH, EHLO, ETRN, HELP, SAML, SEND, SOML, STARTTLS, and VRFY. Along with the support for seven RFC 821 commands (DATA, HELO, MAIL, NOOP, QUIT, RCPT, RSET), the ASA supports a total of fifteen SMTP commands.

Other extended SMTP commands, such as ATRN, ONEX, VERB, CHUNKING, and private extensions and are not supported. Unsupported commands are translated into Xs, which are rejected by the internal server. This results in a message such as “500 Command unknown: 'XXX'.” Incomplete commands are discarded.

The ESMTP inspection engine changes the characters in the server SMTP banner to asterisks except for the “2”, “0”, “0” characters. Carriage return (CR) and linefeed (LF) characters are ignored.

With SMTP inspection enabled, a Telnet session used for interactive SMTP may hang if the following rules are not observed: SMTP commands must be at least four characters in length; must be terminated with carriage return and line feed; and must wait for a response before issuing the next reply.

An SMTP server responds to client requests with numeric reply codes and optional human-readable strings. SMTP application inspection controls and reduces the commands that the user can use as well as the messages that the server returns. SMTP inspection performs three primary tasks:

- Restricts SMTP requests to seven basic SMTP commands and eight extended commands.
- Monitors the SMTP command-response sequence.
- Generates an audit trail—Audit record 108002 is generated when an invalid character embedded in the mail address is replaced. For more information, see RFC 821.

SMTP inspection monitors the command and response sequence for the following anomalous signatures:

- Truncated commands.
- Incorrect command termination (not terminated with <CR><LR>).
- The MAIL and RCPT commands specify who are the sender and the receiver of the mail. Mail addresses are scanned for strange characters. The pipeline character (|) is deleted (changed to a blank space) and “<” ,”>” are only allowed if they are used to define a mail address (“>” must be preceded by “<”).
- Unexpected transition by the SMTP server.
- For unknown commands, the ASA changes all the characters in the packet to X. In this case, the server generates an error code to the client. Because of the change in the packet, the TCP checksum has to be recalculated or adjusted.
- TCP stream editing.

- Command pipelining.

Defaults for ESMTP Inspection

ESMTP inspection is enabled by default, using the _default_esmtp_map inspection policy map.

- The server banner is masked.
- Encrypted connections are not allowed. The STARTTLS indication is removed from the session connection attempt, forcing the client and server to negotiate a plain text session, which can be inspected.
- Special characters in sender and receiver address are not noticed, no action is taken.
- Connections with command line length greater than 512 are dropped and logged.
- Connections with more than 100 recipients are dropped and logged.
- Messages with body length greater than 998 bytes are logged.
- Connections with header line length greater than 998 are dropped and logged.
- Messages with MIME filenames greater than 255 characters are dropped and logged.
- EHLO reply parameters matching “others” are masked.

Following is the policy map configuration:

```
policy-map type inspect esmtp _default_esmtp_map
 description Default ESMTP policy-map
 parameters
  mask-banner
  no mail-relay
  no special-character
  no allow-tls
 match cmd line length gt 512
  drop-connection log
 match cmd RCPT count gt 100
  drop-connection log
 match body line length gt 998
  log
 match header line length gt 998
  drop-connection log
 match sender-address length gt 320
  drop-connection log
 match MIME filename length gt 255
  drop-connection log
 match ehlo-reply-parameter others
  mask
```

Configure ESMTP Inspection

ESMTP inspection is enabled by default. You need to configure it only if you want to different process than that provided by the default inspection map.

Procedure

Step 1 Configure an ESMTP Inspection Policy Map, page 13-42.

Step 2 Configure the ESMTP Inspection Service Policy, page 13-44.

Configure an ESMTP Inspection Policy Map

To specify actions when a message violates a parameter, create an ESMTP inspection policy map. You can then apply the inspection policy map when you enable ESMTP inspection.

Before You Begin

Some traffic matching options use regular expressions for matching purposes. If you intend to use one of those techniques, first create the regular expression or regular expression class map.

Procedure

Step 1 Create an ESMTP inspection policy map, enter the following command:

```
hostname(config)# policy-map type inspect esmtp policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 2 (Optional) To add a description to the policy map, enter the following command:

```
hostname(config-pmap)# description string
```

Step 3 To apply actions to matching traffic, perform the following steps.

a. Specify the traffic on which you want to perform actions using one of the following **match** commands. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

- **match** [**not**] **body** {**length** | **line length**} **gt** *bytes*—Matches messages where the length or length of a line in an ESMTP body message is greater than the specified number of bytes.
- **match** [**not**] **cmd verb** *verb1* [*verb2...*]—Matches the command verb in the message. You can specify one or more of the following commands: auth, data, ehlo, etrn, helo, help, mail, noop, quit, rcpt, rset, saml, soml, vrfy.
- **match** [**not**] **cmd line length gt** *bytes*—Matches messages where the length of a line in the command verb is greater than the specified number of bytes.
- **match** [**not**] **cmd rcpt count gt** *count*—Matches messages where the number of recipients is greater than the specified count.
- **match** [**not**] **ehlo-reply-parameter** *parameter* [*parameter2...*]—Matches ESMTP EHLO reply parameters. You can specify one or more of the following parameters: 8bitmime, auth, binaryname, checkpoint, dsn, etrn, others, pipelining, size, vrfy.

- **match** [**not**] **header** {**length** | **line length**} **gt** *bytes*—Matches messages where the length or length of a line in an ESMTP header is greater than the specified number of bytes.
- **match** [**not**] **header to-fields count gt** *count*—Matches messages where the number of To fields in the header is greater than the specified number.
- **match** [**not**] **invalid-recipients count gt** *number*—Matches messages where the number of invalid recipients is greater than the specified count.
- **match** [**not**] **mime filetype regex** {*regex_name* | **class** *class_name*}—Matches the MIME or media file type against the specified regular expression or regular expression class.
- **match** [**not**] **mime filename length gt** *bytes*—Matches messages where a file name is longer than the specified number of bytes.
- **match** [**not**] **mime encoding** *type* [*type2...*]—Matches the MIME encoding type. You can specify one or more of the following types: 7bit, 8bit, base64, binary, others, quoted-printable.
- **match** [**not**] **sender-address regex** {*regex_name* | **class** *class_name*}—Matches the sender email address against the specified regular expression or regular expression class.
- **match** [**not**] **sender-address length gt** *bytes*—Matches messages where the sender address is greater than the specified number of bytes.

b. Specify the action you want to perform on the matching traffic by entering the following command:

```
hostname(config-pmap-c)# {drop-connection [log]| mask [log] | reset [log] | log |
rate-limit message_rate}
```

Not all options are available for each **match** command. See the CLI help or the command reference for the exact options available.

- The **drop-connection** keyword drops the packet and closes the connection.
- The **mask** keyword masks out the matching portion of the packet. This action is available for **ehlo-reply-parameter** and **cmd verb** only.
- The **reset** keyword drops the packet, closes the connection, and sends a TCP reset to the server and/or client.
- The **log** keyword, which you can use alone or with one of the other keywords, sends a system log message.
- The **rate-limit** *message_rate* argument limits the rate of messages. This option is available with **cmd verb** only, where you can use it as the only action, or you can use it in conjunction with the **mask** action.

You can specify multiple **match** commands in the policy map. For information about the order of **match** commands, see How Multiple Traffic Classes are Handled, page 12-4.

Step 4 To configure parameters that affect the inspection engine, perform the following steps:

a. To enter parameters configuration mode, enter the following command:

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b. Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- **mail-relay** *domain-name* **action** {**drop-connection** [**log**] | **log**}—Identifies a domain name for mail relay. You can either drop the connection and optionally log it, or log it.
- **mask-banner**—Masks the banner from the ESMTP server.

- **special-character action** {**drop-connection** [**log**] | **log**}—Identifies the action to take for messages that include the special characters pipe (|), back quote, and NUL in the sender or receiver email addresses. You can either drop the connection and optionally log it, or log it.
- **allow-tls** [**action log**]—Whether to allow ESMTP over TLS (encrypted connections) without inspection. You can optionally log encrypted connections. The default is **no allow-tls**, which strips the STARTTLS indication from the session connection and forces a plain-text connection.

Example

The following example shows how to define an ESMTP inspection policy map.

```
hostname(config)# regex user1 "user1@cisco.com"
hostname(config)# regex user2 "user2@cisco.com"
hostname(config)# regex user3 "user3@cisco.com"
hostname(config)# class-map type regex senders_black_list
hostname(config-cmap)# description "Regular expressions to filter out undesired senders"
hostname(config-cmap)# match regex user1
hostname(config-cmap)# match regex user2
hostname(config-cmap)# match regex user3

hostname(config)# policy-map type inspect esmtp advanced_esmtp_map
hostname(config-pmap)# match sender-address regex class senders_black_list
hostname(config-pmap-c)# drop-connection log

hostname(config)# policy-map outside_policy
hostname(config-pmap)# class inspection_default
hostname(config-pmap-c)# inspect esmtp advanced_esmtp_map

hostname(config)# service-policy outside_policy interface outside
```

Configure the ESMTP Inspection Service Policy

The default ASA configuration includes ESMTP inspection applied globally on all interfaces. A common method for customizing the inspection configuration is to customize the default global policy. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

class-map *name*
match *parameter*

Example:

```
hostname(config)# class-map esmtp_class_map
hostname(config-cmap)# match access-list esmtp
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

policy-map *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for ESMTP inspection.

class *name*

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure ESMTP inspection.

inspect esmtp [*esmtp_policy_map*]

Where *esmtp_policy_map* is the optional ESMTP inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the ESMTP inspection policy map, see Configure the ESMTP Inspection Service Policy, page 13-44.

Example:

```
hostname(config-class)# no inspect esmtp
hostname(config-class)# inspect esmtp esmtp-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different inspection policy map, you must remove the ESMTP inspection with the **no inspect esmtp** command, and then re-add it with the new inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

TFTP Inspection

TFTP inspection is enabled by default.

TFTP, described in RFC 1350, is a simple protocol to read and write files between a TFTP server and client.

The inspection engine inspects TFTP read request (RRQ), write request (WRQ), and error notification (ERROR), and dynamically creates connections and translations, if necessary, to permit file transfer between a TFTP client and server.

A dynamic secondary channel and a PAT translation, if necessary, are allocated on a reception of a valid read (RRQ) or write (WRQ) request. This secondary channel is subsequently used by TFTP for file transfer or error notification.

Only the TFTP server can initiate traffic over the secondary channel, and at most one incomplete secondary channel can exist between the TFTP client and server. An error notification from the server closes the secondary channel.

TFTP inspection must be enabled if static PAT is used to redirect TFTP traffic.

For information on enabling TFTP inspection, see Configure Application Layer Protocol Inspection, page 12-9.

Inspection for Voice and Video Protocols

The following topics explain application inspection for voice and video protocols. For basic information on why you need to use inspection for certain protocols, and the overall methods for applying inspection, see Getting Started with Application Layer Protocol Inspection, page 12-1.

CTIQBE Inspection

CTIQBE protocol inspection supports NAT, PAT, and bidirectional NAT. This enables Cisco IP SoftPhone and other Cisco TAPI/JTAPI applications to work successfully with Cisco CallManager for call setup across the ASA.

TAPI and JTAPI are used by many Cisco VoIP applications. CTIQBE is used by Cisco TSP to communicate with Cisco CallManager.

For information on enabling CTIQBE inspection, see Configure Application Layer Protocol Inspection, page 12-9.

Limitations for CTIQBE Inspection

Stateful failover of CTIQBE calls is not supported.

The following summarizes special considerations when using CTIQBE application inspection in specific scenarios:

- If two Cisco IP SoftPhones are registered with different Cisco CallManagers, which are connected to different interfaces of the ASA, calls between these two phones fail.

- When Cisco CallManager is located on the higher security interface compared to Cisco IP SoftPhones, if NAT or outside NAT is required for the Cisco CallManager IP address, the mapping must be static as Cisco IP SoftPhone requires the Cisco CallManager IP address to be specified explicitly in its Cisco TSP configuration on the PC.
- When using PAT or Outside PAT, if the Cisco CallManager IP address is to be translated, its TCP port 2748 must be statically mapped to the same port of the PAT (interface) address for Cisco IP SoftPhone registrations to succeed. The CTIQBE listening port (TCP 2748) is fixed and is not user-configurable on Cisco CallManager, Cisco IP SoftPhone, or Cisco TSP.

Verifying and Monitoring CTIQBE Inspection

The **show ctiqbe** command displays information regarding the CTIQBE sessions established across the ASA. It shows information about the media connections allocated by the CTIQBE inspection engine.

The following is sample output from the **show ctiqbe** command under the following conditions. There is only one active CTIQBE session setup across the ASA. It is established between an internal CTI device (for example, a Cisco IP SoftPhone) at local address 10.0.0.99 and an external Cisco CallManager at 172.29.1.77, where TCP port 2748 is the Cisco CallManager. The heartbeat interval for the session is 120 seconds.

```
hostname# # show ctiqbe

Total: 1
        LOCAL           FOREIGN         STATE   HEARTBEAT
-----------------------------------------------------------
1       10.0.0.99/1117  172.29.1.77/2748        1       120
        ------------------------------------------
        RTP/RTCP: PAT xlates: mapped to 172.29.1.99(1028 - 1029)
        ------------------------------------------
        MEDIA: Device ID 27     Call ID 0
               Foreign 172.29.1.99      (1028 - 1029)
               Local   172.29.1.88      (26822 - 26823)
        ------------------------------------------
```

The CTI device has already registered with the CallManager. The device internal address and RTP listening port is PATed to 172.29.1.99 UDP port 1028. Its RTCP listening port is PATed to UDP 1029.

The line beginning with `RTP/RTCP: PAT xlates:` appears only if an internal CTI device has registered with an external CallManager and the CTI device address and ports are PATed to that external interface. This line does not appear if the CallManager is located on an internal interface, or if the internal CTI device address and ports are translated to the same external interface that is used by the CallManager.

The output indicates a call has been established between this CTI device and another phone at 172.29.1.88. The RTP and RTCP listening ports of the other phone are UDP 26822 and 26823. The other phone locates on the same interface as the CallManager because the ASA does not maintain a CTIQBE session record associated with the second phone and CallManager. The active call leg on the CTI device side can be identified with Device ID 27 and Call ID 0.

The following is sample output from the **show xlate debug** command for these CTIBQE connections:

```
hostname# show xlate debug
3 in use, 3 most used
Flags:  D - DNS, d - dump, I - identity, i - inside, n - no random,
        r - portmap, s - static
TCP PAT from inside:10.0.0.99/1117 to outside:172.29.1.99/1025 flags ri idle 0:00:22
timeout 0:00:30
UDP PAT from inside:10.0.0.99/16908 to outside:172.29.1.99/1028 flags ri idle 0:00:00
timeout 0:04:10
```

```
UDP PAT from inside:10.0.0.99/16909 to outside:172.29.1.99/1029 flags ri idle 0:00:23
timeout 0:04:10
```

The **show conn state ctiqbe** command displays the status of CTIQBE connections. In the output, the media connections allocated by the CTIQBE inspection engine are denoted by a ‘C’ flag. The following is sample output from the **show conn state ctiqbe** command:

```
hostname# show conn state ctiqbe
1 in use, 10 most used
hostname# show conn state ctiqbe detail
1 in use, 10 most used
Flags: A - awaiting inside ACK to SYN, a - awaiting outside ACK to SYN,
       B - initial SYN from outside, C - CTIQBE media, D - DNS, d - dump,
       E - outside back connection, F - outside FIN, f - inside FIN,
       G - group, g - MGCP, H - H.323, h - H.225.0, I - inbound data,
       i - incomplete, J - GTP, j - GTP data, k - Skinny media,
       M - SMTP data, m - SIP media, O - outbound data, P - inside back connection,
       q - SQL*Net data, R - outside acknowledged FIN,
       R - UDP RPC, r - inside acknowledged FIN, S - awaiting inside SYN,
       s - awaiting outside SYN, T - SIP, t - SIP transient, U - up
```

H.323 Inspection

The following sections describe the H.323 application inspection.

H.323 Inspection Overview

H.323 inspection provides support for H.323 compliant applications such as Cisco CallManager and VocalTec Gatekeeper. H.323 is a suite of protocols defined by the International Telecommunication Union for multimedia conferences over LANs. The ASA supports H.323 through Version 6, including H.323 v3 feature Multiple Calls on One Call Signaling Channel.

With H.323 inspection enabled, the ASA supports multiple calls on the same call signaling channel, a feature introduced with H.323 Version 3. This feature reduces call setup time and reduces the use of ports on the ASA.

The two major functions of H.323 inspection are as follows:

- NAT the necessary embedded IPv4 addresses in the H.225 and H.245 messages. Because H.323 messages are encoded in PER encoding format, the ASA uses an ASN.1 decoder to decode the H.323 messages.
- Dynamically allocate the negotiated H.245 and RTP/RTCP connections. The H.225 connection can also be dynamically allocated when using RAS.

How H.323 Works

The H.323 collection of protocols collectively may use up to two TCP connection and four to eight UDP connections. FastConnect uses only one TCP connection, and RAS uses a single UDP connection for registration, admissions, and status.

An H.323 client can initially establish a TCP connection to an H.323 server using TCP port 1720 to request Q.931 call setup. As part of the call setup process, the H.323 terminal supplies a port number to the client to use for an H.245 TCP connection. In environments where H.323 gatekeeper is in use, the initial packet is transmitted using UDP.

H.323 inspection monitors the Q.931 TCP connection to determine the H.245 port number. If the H.323 terminals are not using FastConnect, the ASA dynamically allocates the H.245 connection based on the inspection of the H.225 messages.

Note The H.225 connection can also be dynamically allocated when using RAS.

Within each H.245 message, the H.323 endpoints exchange port numbers that are used for subsequent UDP data streams. H.323 inspection inspects the H.245 messages to identify these ports and dynamically creates connections for the media exchange. RTP uses the negotiated port number, while RTCP uses the next higher port number.

The H.323 control channel handles H.225 and H.245 and H.323 RAS. H.323 inspection uses the following ports.

- 1718—Gate Keeper Discovery UDP port
- 1719—RAS UDP port
- 1720—TCP Control Port

You must permit traffic for the well-known H.323 port 1719 for RAS signaling. Additionally, you must permit traffic for the well-known H.323 port 1720 for the H.225 call signaling; however, the H.245 signaling ports are negotiated between the endpoints in the H.225 signaling. When an H.323 gatekeeper is used, the ASA opens an H.225 connection based on inspection of the ACF and RCF messages.

After inspecting the H.225 messages, the ASA opens the H.245 channel and then inspects traffic sent over the H.245 channel as well. All H.245 messages passing through the ASA undergo H.245 application inspection, which translates embedded IP addresses and opens the media channels negotiated in H.245 messages.

The H.323 ITU standard requires that a TPKT header, defining the length of the message, precede the H.225 and H.245, before being passed on to the reliable connection. Because the TPKT header does not necessarily need to be sent in the same TCP packet as H.225 and H.245 messages, the ASA must remember the TPKT length to process and decode the messages properly. For each connection, the ASA keeps a record that contains the TPKT length for the next expected message.

If the ASA needs to perform NAT on IP addresses in messages, it changes the checksum, the UUIE length, and the TPKT, if it is included in the TCP packet with the H.225 message. If the TPKT is sent in a separate TCP packet, the ASA proxy ACKs that TPKT and appends a new TPKT to the H.245 message with the new length.

Note The ASA does not support TCP options in the Proxy ACK for the TPKT.

Each UDP connection with a packet going through H.323 inspection is marked as an H.323 connection and times out with the H.323 timeout as configured with the **timeout** command.

Note You can enable call setup between H.323 endpoints when the Gatekeeper is inside the network. The ASA includes options to open pinholes for calls based on the RegistrationRequest/RegistrationConfirm (RRQ/RCF) messages. Because these RRQ/RCF messages are sent to and from the Gatekeeper, the calling endpoint's IP address is unknown and the ASA opens a pinhole through source IP address/port 0/0. By default, this option is disabled. To enable call setup between H.323 endpoint, enter the **ras-rcf-pinholes enable** command during parameter configuration mode while creating an H.323 Inspection policy map. See Configure H.323 Inspection Policy Map, page 14-6.

H.239 Support in H.245 Messages

The ASA sits between two H.323 endpoints. When the two H.323 endpoints set up a telepresentation session so that the endpoints can send and receive a data presentation, such as spreadsheet data, the ASA ensure successful H.239 negotiation between the endpoints.

H.239 is a standard that provides the ability for H.300 series endpoints to open an additional video channel in a single call. In a call, an endpoint (such as a video phone), sends a channel for video and a channel for data presentation. The H.239 negotiation occurs on the H.245 channel.

The ASA opens pinholes for the additional media channel and the media control channel. The endpoints use open logical channel message (OLC) to signal a new channel creation. The message extension is part of H.245 version 13.

The decoding and encoding of the telepresentation session is enabled by default. H.239 encoding and decoding is preformed by ASN.1 coder.

Limitations for H.323 Inspection

H.323 inspection is tested and supported for Cisco Unified Communications Manager (CUCM) 7.0. It is not supported for CUCM 8.0 and higher. H.323 inspection might work with other releases and products.

The following are some of the known issues and limitations when using H.323 application inspection:

- Only static NAT is fully supported. Static PAT may not properly translate IP addresses embedded in optional fields within H.323 messages. If you experience this kind of problem, do not use static PAT with H.323.
- Not supported with dynamic NAT or PAT.
- Not supported with extended PAT.
- Not supported with NAT between same-security-level interfaces.
- Not supported with outside NAT.
- Not supported with NAT64.
- When a NetMeeting client registers with an H.323 gatekeeper and tries to call an H.323 gateway that is also registered with the H.323 gatekeeper, the connection is established but no voice is heard in either direction. This problem is unrelated to the ASA.
- If you configure a network static address where the network static address is the same as a third-party netmask and address, then any outbound H.323 connection fails.

Configure H.323 Inspection

H.323 inspection supports RAS, H.225, and H.245, and its functionality translates all embedded IP addresses and ports. It performs state tracking and filtering and can do a cascade of inspect function activation. H.323 inspection supports phone number filtering, dynamic T.120 control, H.245 tunneling control, HSI groups, protocol state tracking, H.323 call duration enforcement, and audio/video control.

H.323 inspection is enabled by default. You need to configure it only if you want non-default processing. If you want to customize H.323 inspection, use the following process.

Procedure

Step 1 Configure H.323 Inspection Policy Map, page 14-6

Step 2 Configure the H.323 Inspection Service Policy, page 14-9

Configure H.323 Inspection Policy Map

You can create an H.323 inspection policy map to customize H.323 inspection actions if the default inspection behavior is not sufficient for your network.

When defining traffic matching criteria, you can either create a class map or include the match statements directly in the policy map. The following procedure explains both approaches.

Before You Begin

Some traffic matching options use regular expressions for matching purposes. If you intend to use one of those techniques, first create the regular expression or regular expression class map.

Procedure

Step 1 (Optional) Create an H.323 inspection class map by performing the following steps.

A class map groups multiple traffic matches.You can alternatively identify **match** commands directly in the policy map. The difference between creating a class map and defining the traffic match directly in the inspection policy map is that the class map lets you create more complex match criteria, and you can reuse class maps.

To specify traffic that should not match the class map, use the **match not** command. For example, if the **match not** command specifies the string “example.com,” then any traffic that includes “example.com” does not match the class map.

For the traffic that you identify in this class map, you specify actions to take on the traffic in the inspection policy map.

If you want to perform different actions for each **match** command, you should identify the traffic directly in the policy map.

a. Create the class map by entering the following command:

```
hostname(config)# class-map type inspect h323 [match-all | match-any] class_map_name
hostname(config-cmap)#
```

Where *class_map_name* is the name of the class map. The **match-all** keyword is the default, and specifies that traffic must match all criteria to match the class map. The **match-any** keyword specifies that the traffic matches the class map if it matches at least one of the criteria. The CLI enters class-map configuration mode, where you can enter one or more **match** commands.

b. (Optional) To add a description to the class map, enter the following command:

```
hostname(config-cmap)# description string
```

Where *string* is the description of the class map (up to 200 characters).

c. Specify the traffic on which you want to perform actions using one of the following **match** commands. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

- **match** [**not**] **called-party regex** {*regex_name* | **class** *class_name*}—Matches the called party against the specified regular expression or regular expression class.
- **match** [**not**] **calling-party regex** {*regex_name* | **class** *class_name*}—Matches the calling party against the specified regular expression or regular expression class.
- **match** [**not**] **media-type** {**audio** | **data** | **video**}—Matches the media type.

Step 2 Create an H.323 inspection policy map:

```
hostname(config)# policy-map type inspect h323 policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 3 (Optional) To add a description to the policy map, enter the following command:

```
hostname(config-pmap)# description string
```

Step 4 To apply actions to matching traffic, perform the following steps.

You can specify multiple **class** or **match** commands in the policy map. For information about the order of **class** and **match** commands, see How Multiple Traffic Classes are Handled, page 12-4.

a. Specify the traffic on which you want to perform actions using one of the following methods:

- If you created an H.323 class map, specify it by entering the following command:

```
hostname(config-pmap)# class class_map_name
hostname(config-pmap-c)#
```

- Specify traffic directly in the policy map using one of the **match** commands described for H.323 class maps. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

b. Specify the action you want to perform on the matching traffic by entering the following command:

```
hostname(config-pmap-c)# {drop [log] | drop-connection | reset}
```

The **drop** keyword drops the packet. For media type matches, you can include the **log** keyword to send a system log message.

The **drop-connection** keyword drops the packet and closes the connection. This option is available for called or calling party matching.

The **reset** keyword drops the packet, closes the connection, and sends a TCP reset to the server and/or client. This option is available for called or calling party matching.

Step 5 To configure parameters that affect the inspection engine, perform the following steps:

a. To enter parameters configuration mode, enter the following command:

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b. Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- **ras-rcf-pinholes enable**—Enables call setup between H.323 endpoints. You can enable call setup between H.323 endpoints when the Gatekeeper is inside the network. Use this option to open pinholes for calls based on the RegistrationRequest/RegistrationConfirm (RRQ/RCF) messages. Because these RRQ/RCF messages are sent to and from the Gatekeeper, the calling endpoint's IP address is unknown and the ASA opens a pinhole through source IP address/port 0/0. By default, this option is disabled.
- **timeout users** *time*—Sets the H.323 call duration limit (in hh:mm:ss format). To have no timeout, specify 00:00:00. Range is from 0:0:0 to 1193:0;0.
- **call-party-number**—Enforces sending call party number during call setup.
- **h245-tunnel-block action** {**drop-connection** | **log**}—Enforces H.245 tunnel blocking. Specify whether you want to drop the connection or simply log it.
- **rtp-conformance** [**enforce-payloadtype**]—Checks RTP packets flowing on the pinholes for protocol conformance. The optional **enforce-payloadtype** keyword enforces the payload type to be audio or video based on the signaling exchange.
- **state-checking** {**h225** | **ras**}—Enables state checking validation. You can enter the command separately to enable state checking for H.225 and RAS.

Step 6 While still in parameter configuration mode, you can configure HSI groups.

a. Define an HSI group and enter HSI group configuration mode.

```
hostname(config-pmap-p)# hsi-group id
```

Where *id* is the HSI group ID. Range is from 0 to 2147483647.

b. Add an HSI to the HSI group using the IP address. You can add a maximum of five hosts per HSI group.

```
hostname(config-h225-map-hsi-grp)# hsi ip_address
```

c. Add an endpoint to the HSI group.

```
hostname(config-h225-map-hsi-grp)# endpoint ip_address if_name
```

Where *ip_address* is the endpoint to add and *if_name* is the interface through which the endpoint is connected to the ASA. You can add a maximum of ten endpoints per HSI group.

Example

The following example shows how to configure phone number filtering:

```
hostname(config)# regex caller 1 "5551234567"
hostname(config)# regex caller 2 "5552345678"
hostname(config)# regex caller 3 "5553456789"

hostname(config)# class-map type inspect h323 match-all h323_traffic
hostname(config-pmap-c)# match called-party regex caller1
hostname(config-pmap-c)# match calling-party regex caller2

hostname(config)# policy-map type inspect h323 h323_map
hostname(config-pmap)# parameters
hostname(config-pmap-p)# class h323_traffic
```

```
hostname(config-pmap-c)# drop
```

Configure the H.323 Inspection Service Policy

The default ASA configuration includes H.323 H.255 and RAS inspection on the default ports applied globally on all interfaces. A common method for customizing the inspection configuration is to customize the default global policy. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

class-map *name*
match *parameter*

Example:

```
hostname(config)# class-map h323_class_map
hostname(config-cmap)# match access-list h323
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

policy-map *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for H.323 inspection.

class *name*

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure H.323 inspection.

inspect h323 {**h255** | **ras**} [*h323_policy_map*]

Where *h323_policy_map* is the optional H.323 inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the H.323 inspection policy map, see Configure H.323 Inspection Policy Map, page 14-6.

Example:

```
hostname(config-class)# no inspect h323 h225
hostname(config-class)# no inspect h323 ras
```

```
hostname(config-class)# inspect h255 h323-map
hostname(config-class)# inspect ras h323-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different H.323 inspection policy map, you must remove the H.323 inspection with the **no inspect h323** command, and then re-add it with the new H.323 inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Configuring H.323 and H.225 Timeout Values

You can configure H.323/H.255 global timeout values on the **Configuration > Firewall > Advanced > Global Timeouts** page. You can set the interval for inactivity after which an H.255 signaling connection is closed (default is 1 hour) or an H.323 control connection is closed (default is 5 minutes).

To configure the idle time after which an H.225 signaling connection is closed, use the **timeout h225** command. The default for H.225 timeout is one hour.

To configure the idle time after which an H.323 control connection is closed, use the **timeout h323** command. The default is five minutes.

Verifying and Monitoring H.323 Inspection

The following sections describe how to display information about H.323 sessions.

- Monitoring H.225 Sessions, page 14-10
- Monitoring H.245 Sessions, page 14-11
- Monitoring H.323 RAS Sessions, page 14-12

Monitoring H.225 Sessions

The **show h225** command displays information for H.225 sessions established across the ASA. Along with the **debug h323 h225 event**, **debug h323 h245 event**, and **show local-host** commands, this command is used for troubleshooting H.323 inspection engine issues.

If there is an abnormally large number of connections, check that the sessions are timing out based on the default timeout values or the values set by you. If they are not, then there is a problem that needs to be investigated.

The following is sample output from the **show h225** command:

```
hostname# show h225
Total H.323 Calls: 1
1 Concurrent Call(s) for
    Local:   10.130.56.3/1040   Foreign: 172.30.254.203/1720
    1. CRV 9861
    Local:   10.130.56.3/1040   Foreign: 172.30.254.203/1720
0 Concurrent Call(s) for
    Local:   10.130.56.4/1050   Foreign: 172.30.254.205/1720
```

This output indicates that there is currently 1 active H.323 call going through the ASA between the local endpoint 10.130.56.3 and foreign host 172.30.254.203, and for these particular endpoints, there is 1 concurrent call between them, with a CRV for that call of 9861.

For the local endpoint 10.130.56.4 and foreign host 172.30.254.205, there are 0 concurrent calls. This means that there is no active call between the endpoints even though the H.225 session still exists. This could happen if, at the time of the **show h225** command, the call has already ended but the H.225 session has not yet been deleted. Alternately, it could mean that the two endpoints still have a TCP connection opened between them because they set "maintainConnection" to TRUE, so the session is kept open until they set it to FALSE again, or until the session times out based on the H.225 timeout value in your configuration.

Monitoring H.245 Sessions

The **show h245** command displays information for H.245 sessions established across the ASA by endpoints using slow start. Slow start is when the two endpoints of a call open another TCP control channel for H.245. Fast start is where the H.245 messages are exchanged as part of the H.225 messages on the H.225 control channel.)

The following is sample output from the **show h245** command:

```
hostname# show h245
Total: 1
        LOCAL           TPKT    FOREIGN         TPKT
1       10.130.56.3/1041        0       172.30.254.203/1245    0
        MEDIA: LCN 258 Foreign 172.30.254.203 RTP 49608 RTCP 49609
                       Local   10.130.56.3 RTP 49608 RTCP 49609
        MEDIA: LCN 259 Foreign 172.30.254.203 RTP 49606 RTCP 49607
                       Local   10.130.56.3 RTP 49606 RTCP 49607
```

There is currently one H.245 control session active across the ASA. The local endpoint is 10.130.56.3, and we are expecting the next packet from this endpoint to have a TPKT header because the TPKT value is 0. The TKTP header is a 4-byte header preceding each H.225/H.245 message. It gives the length of the message, including the 4-byte header. The foreign host endpoint is 172.30.254.203, and we are expecting the next packet from this endpoint to have a TPKT header because the TPKT value is 0.

The media negotiated between these endpoints have an LCN of 258 with the foreign RTP IP address/port pair of 172.30.254.203/49608 and an RTCP IP address/port of 172.30.254.203/49609 with a local RTP IP address/port pair of 10.130.56.3/49608 and an RTCP port of 49609.

The second LCN of 259 has a foreign RTP IP address/port pair of 172.30.254.203/49606 and an RTCP IP address/port pair of 172.30.254.203/49607 with a local RTP IP address/port pair of 10.130.56.3/49606 and RTCP port of 49607.

Monitoring H.323 RAS Sessions

The **show h323 ras** command displays connection information for H.323 RAS sessions established across the ASA between a gatekeeper and its H.323 endpoint. Along with the **debug h323 ras event** and **show local-host** commands, this command is used for troubleshooting H.323 RAS inspection engine issues.

The following is sample output from the **show h323 ras** command:

```
hostname# show h323 ras
Total: 1
        GK                      Caller
        172.30.254.214 10.130.56.14
```

This output shows that there is one active registration between the gatekeeper 172.30.254.214 and its client 10.130.56.14.

MGCP Inspection

The following sections describe MGCP application inspection.

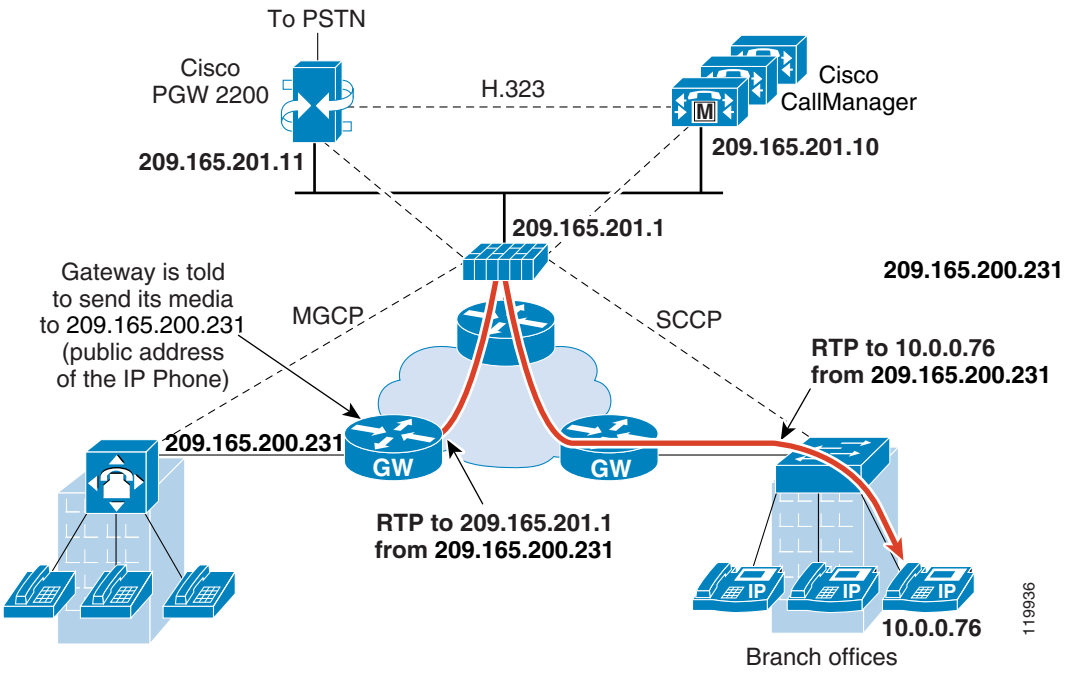
MGCP Inspection Overview

MGCP is a master/slave protocol used to control media gateways from external call control elements called media gateway controllers or call agents. A media gateway is typically a network element that provides conversion between the audio signals carried on telephone circuits and data packets carried over the Internet or over other packet networks. Using NAT and PAT with MGCP lets you support a large number of devices on an internal network with a limited set of external (global) addresses. Examples of media gateways are:

- Trunking gateways, that interface between the telephone network and a Voice over IP network. Such gateways typically manage a large number of digital circuits.
- Residential gateways, that provide a traditional analog (RJ11) interface to a Voice over IP network. Examples of residential gateways include cable modem/cable set-top boxes, xDSL devices, broad-band wireless devices.
- Business gateways, that provide a traditional digital PBX interface or an integrated soft PBX interface to a Voice over IP network.

MGCP messages are transmitted over UDP. A response is sent back to the source address (IP address and UDP port number) of the command, but the response may not arrive from the same address as the command was sent to. This can happen when multiple call agents are being used in a failover configuration and the call agent that received the command has passed control to a backup call agent, which then sends the response. The following figure illustrates how you can use NAT with MGCP.

Figure 14-1 Using NAT with MGCP



MGCP endpoints are physical or virtual sources and destinations for data. Media gateways contain endpoints on which the call agent can create, modify and delete connections to establish and control media sessions with other multimedia endpoints. Also, the call agent can instruct the endpoints to detect certain events and generate signals. The endpoints automatically communicate changes in service state to the call agent.

- Gateways usually listen to UDP port 2427 to receive commands from the call agent.
- The port on which the call agent receives commands from the gateway. Call agents usually listen to UDP port 2727 to receive commands from the gateway.

Note MGCP inspection does not support the use of different IP addresses for MGCP signaling and RTP data. A common and recommended practice is to send RTP data from a resilient IP address, such as a loopback or virtual IP address; however, the ASA requires the RTP data to come from the same address as MGCP signaling.

Configure MGCP Inspection

Use the following process to enable MGCP inspection.

Procedure

Step 1 Configuring an MGCP Inspection Policy Map for Additional Inspection Control, page 14-14.

Step 2 Configure the MGCP Inspection Service Policy, page 14-15.

Configuring an MGCP Inspection Policy Map for Additional Inspection Control

If the network has multiple call agents and gateways for which the ASA has to open pinholes, create an MGCP map. You can then apply the MGCP map when you enable MGCP inspection.

Procedure

Step 1 To create an MGCP inspection policy map, enter the following command:

```
hostname(config)# policy-map type inspect mgcp map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 2 (Optional) To add a description to the policy map, enter the following command:

```
hostname(config-pmap)# description string
```

Step 3 Enter parameters configuration mode.

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

Step 4 Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option.

- **call-agent** *ip_address group_id*—Configures the call agent groups that can manage one or more gateways. The call agent group information is used to open connections for the call agents in the group (other than the one a gateway sends a command to) so that any of the call agents can send the response. Call agents with the same *group_id* belong to the same group. A call agent may belong to more than one group. The *group_id* option is a number from 0 to 4294967295. The *ip_address* option specifies the IP address of the call agent.

  Note MGCP call agents send AUEP messages to determine if MGCP end points are present. This establishes a flow through the ASA and allows MGCP end points to register with the call agent.

- **gateway** *ip_address group_id*—Identifies which group of call agents is managing a particular gateway. The IP address of the gateway is specified with the *ip_address* option. The *group_id* option is a number from 0 to 4294967295 that must correspond with the *group_id* of the call agents that are managing the gateway. A gateway may only belong to one group.
- **command-queue** *command_limit*—Sets the maximum number of commands allowed in the MGCP command queue, from 1 to 2147483647. The default is 200.

Example

The following example shows how to define an MGCP map:

```
hostname(config)# policy-map type inspect mgcp sample_map
hostname(config-pmap)# parameters
hostname(config-pmap-p)# call-agent 10.10.11.5 101
hostname(config-pmap-p)# call-agent 10.10.11.6 101
hostname(config-pmap-p)# call-agent 10.10.11.7 102
hostname(config-pmap-p)# call-agent 10.10.11.8 102
hostname(config-pmap-p)# gateway 10.10.10.115 101
```

```
hostname(config-pmap-p)# gateway 10.10.10.116 102
hostname(config-pmap-p)# gateway 10.10.10.117 102
hostname(config-pmap-p)# command-queue 150
```

Configure the MGCP Inspection Service Policy

MGCP inspection is not enabled in the default inspection policy, so you must enable it if you need this inspection. However, the default inspect class does include the default MGCP ports, so you can simply edit the default global inspection policy to add MGCP inspection. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

class-map *name*
match *parameter*

Example:

```
hostname(config)# class-map mgcp_class_map
hostname(config-cmap)# match access-list mgcp
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

policy-map *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for MGCP inspection.

class *name*

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure MGCP inspection.

inspect mgcp [*mgcp_policy_map*]

Where *mgcp_policy_map* is the optional MGCP inspection policy map. For information on creating the MGCP inspection policy map, see Configuring an MGCP Inspection Policy Map for Additional Inspection Control, page 14-14.

Example:

```
hostname(config-class)# no inspect mgcp
hostname(config-class)# inspect mgcp mgcp-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different MGCP inspection policy map, you must remove the MGCP inspection with the **no inspect mgcp** command, and then re-add it with the new MGCP inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Configuring MGCP Timeout Values

You can configure several MGCP global timeout values on the **Configuration > Firewall > Advanced > Global Timeouts** page. You can set the interval for inactivity after which an MGCP media connection is closed (default is 5 minutes). You can also set the timeout for PAT xlates (30 seconds).

The **timeout mgcp command** lets you set the interval for inactivity after which an MGCP media connection is closed. The default is 5 minutes.

The **timeout mgcp-pat** command lets you set the timeout for PAT xlates. Because MGCP does not have a keepalive mechanism, if you use non-Cisco MGCP gateways (call agents), the PAT xlates are torn down after the default timeout interval, which is 30 seconds.

Verifying and Monitoring MGCP Inspection

The **show mgcp commands** command lists the number of MGCP commands in the command queue. The **show mgcp sessions** command lists the number of existing MGCP sessions. The **detail** option includes additional information about each command (or session) in the output. The following is sample output from the **show mgcp commands** command:

```
hostname# show mgcp commands
1 in use, 1 most used, 200 maximum allowed
CRCX, gateway IP: host-pc-2, transaction ID: 2052, idle: 0:00:07
```

The following is sample output from the **show mgcp detail** command.

```
hostname# show mgcp commands detail
1 in use, 1 most used, 200 maximum allowed
CRCX, idle: 0:00:10
        Gateway IP      host-pc-2
        Transaction ID  2052
        Endpoint name   aaln/1
        Call ID         9876543210abcdef
        Connection ID
```

```
        Media IP        192.168.5.7
        Media port      6058
```

The following is sample output from the **show mgcp sessions** command.

```
hostname# show mgcp sessions
1 in use, 1 most used
Gateway IP host-pc-2, connection ID 6789af54c9, active 0:00:11
```

The following is sample output from the **show mgcp sessions detail** command.

```
hostname# show mgcp sessions detail
1 in use, 1 most used
Session active 0:00:14
        Gateway IP      host-pc-2
        Call ID         9876543210abcdef
        Connection ID   6789af54c9
        Endpoint name   aaln/1
        Media lcl port  6166
        Media rmt IP    192.168.5.7
        Media rmt port  6058
```

RTSP Inspection

The following sections describe RTSP application inspection.

RTSP Inspection Overview

The RTSP inspection engine lets the ASA pass RTSP packets. RTSP is used by RealAudio, RealNetworks, Apple QuickTime 4, RealPlayer, and Cisco IP/TV connections.

Note For Cisco IP/TV, use RTSP TCP ports 554 and 8554.

RTSP applications use the well-known port 554 with TCP (rarely UDP) as a control channel. The ASA only supports TCP, in conformity with RFC 2326. This TCP control channel is used to negotiate the data channels that are used to transmit audio/video traffic, depending on the transport mode that is configured on the client.

The supported RDT transports are: rtp/avp, rtp/avp/udp, x-real-rdt, x-real-rdt/udp, and x-pn-tng/udp.

The ASA parses Setup response messages with a status code of 200. If the response message is traveling inbound, the server is outside relative to the ASA and dynamic channels need to be opened for connections coming inbound from the server. If the response message is outbound, then the ASA does not need to open dynamic channels.

Because RFC 2326 does not require that the client and server ports must be in the SETUP response message, the ASA keeps state and remembers the client ports in the SETUP message. QuickTime places the client ports in the SETUP message and then the server responds with only the server ports.

RTSP inspection does not support PAT or dual-NAT. Also, the ASA cannot recognize HTTP cloaking where RTSP messages are hidden in the HTTP messages.

RealPlayer Configuration Requirements

When using RealPlayer, it is important to properly configure transport mode. For the ASA, add an **access-list** command from the server to the client or vice versa. For RealPlayer, change transport mode by clicking **Options>Preferences>Transport>RTSP Settings**.

If using TCP mode on the RealPlayer, select the **Use TCP to Connect to Server** and **Attempt to use TCP for all content** check boxes. On the ASA, there is no need to configure the inspection engine.

If using UDP mode on the RealPlayer, select the **Use TCP to Connect to Server** and **Attempt to use UDP for static content** check boxes, and for live content not available via multicast. On the ASA, add an **inspect rtsp** *port* command.

Limitations for RSTP Inspection

The following restrictions apply to the RSTP inspection.

- The ASA does not support multicast RTSP or RTSP messages over UDP.
- The ASA does not have the ability to recognize HTTP cloaking where RTSP messages are hidden in the HTTP messages.
- The ASA cannot perform NAT on RTSP messages because the embedded IP addresses are contained in the SDP files as part of HTTP or RTSP messages. Packets could be fragmented and the ASA cannot perform NAT on fragmented packets.
- With Cisco IP/TV, the number of translates the ASA performs on the SDP part of the message is proportional to the number of program listings in the Content Manager (each program listing can have at least six embedded IP addresses).
- You can configure NAT for Apple QuickTime 4 or RealPlayer. Cisco IP/TV only works with NAT if the Viewer and Content Manager are on the outside network and the server is on the inside network.

Configure RTSP Inspection

RTSP inspection is enabled by default. You need to configure it only if you want non-default processing. If you want to customize RTSP inspection, use the following process.

Procedure

Step 1 Configure RTSP Inspection Policy Map, page 14-19

Step 2 Configure the RTSP Inspection Service Policy, page 14-21

Configure RTSP Inspection Policy Map

You can create an RTSP inspection policy map to customize RTSP inspection actions if the default inspection behavior is not sufficient for your network.

When defining traffic matching criteria, you can either create a class map or include the match statements directly in the policy map. The following procedure explains both approaches.

Before You Begin

Some traffic matching options use regular expressions for matching purposes. If you intend to use one of those techniques, first create the regular expression or regular expression class map.

Procedure

Step 1 (Optional) Create an RTSP inspection class map by performing the following steps.

A class map groups multiple traffic matches.You can alternatively identify **match** commands directly in the policy map. The difference between creating a class map and defining the traffic match directly in the inspection policy map is that the class map lets you create more complex match criteria, and you can reuse class maps.

To specify traffic that should not match the class map, use the **match not** command. For example, if the **match not** command specifies the string "example.com," then any traffic that includes "example.com" does not match the class map.

For the traffic that you identify in this class map, you specify actions to take on the traffic in the inspection policy map.

If you want to perform different actions for each **match** command, you should identify the traffic directly in the policy map.

a. Create the class map by entering the following command:

```
hostname(config)# class-map type inspect rtsp [match-all | match-any] class_map_name
hostname(config-cmap)#
```

Where *class_map_name* is the name of the class map. The **match-all** keyword is the default, and specifies that traffic must match all criteria to match the class map. The **match-any** keyword specifies that the traffic matches the class map if it matches at least one of the criteria. The CLI enters class-map configuration mode, where you can enter one or more **match** commands.

b. (Optional) To add a description to the class map, enter the following command:

```
hostname(config-cmap)# description string
```

c. Specify the traffic on which you want to perform actions using one of the following **match** commands. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

- **match** [**not**] **request-method** *method*—Matches an RTSP request method. The methods are: announce, describe, get_parameter, options, pause, play, record, redirect, setup, set_parameter, teardown.
- **match** [**not**] **url-filter regex** {*regex_name* | **class** *class_name*}—Matches the URL against the specified regular expression or regular expression class.

Step 2 To create an RTSP inspection policy map, enter the following command:

```
hostname(config)# policy-map type inspect rtsp policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 3 (Optional) To add a description to the policy map, enter the following command:

```
hostname(config-pmap)# description string
```

Step 4 To apply actions to matching traffic, perform the following steps.

a. Specify the traffic on which you want to perform actions using one of the following methods:

- If you created an RTSP class map, specify it by entering the following command:

```
hostname(config-pmap)# class class_map_name
hostname(config-pmap-c)#
```

- Specify traffic directly in the policy map using one of the **match** commands described for RTSP class maps. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

b. Specify the action you want to perform on the matching traffic by entering the following command:

```
hostname(config-pmap-c)# {drop-connection [log] | log | rate-limit message_rate}
```

The **drop-connection** keyword drops the packet and closes the connection. This option is available for URL matching.

The **log** keyword, which you can use alone or with **drop-connection**, sends a system log message.

The **rate-limit** *message_rate* argument limits the rate of messages per second. This option is available for request method matching.

You can specify multiple **class** or **match** commands in the policy map. For information about the order of **class** and **match** commands, see How Multiple Traffic Classes are Handled, page 12-4.

Step 5 To configure parameters that affect the inspection engine, perform the following steps:

a. To enter parameters configuration mode, enter the following command:

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b. Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- **reserve-port-protect**—Restricts the use of reserve ports during media negotiation.
- **url-length-limit** *bytes*—Sets a limit on the URL length allowed in the message, from 0 to 6000 bytes.

Example

The following example shows a how to define an RTSP inspection policy map.

```
hostname(config)# regex badurl1 www.url1.com/rtsp.avi
hostname(config)# regex badurl2 www.url2.com/rtsp.rm
hostname(config)# regex badurl3 www.url3.com/rtsp.asp

hostname(config)# class-map type regex match-any badurl-list
hostname(config-cmap)# match regex badurl1
hostname(config-cmap)# match regex badurl2
hostname(config-cmap)# match regex badurl3

hostname(config)# policy-map type inspect rtsp rtsp-filter-map
hostname(config-pmap)# match url-filter regex class badurl-list
```

```
hostname(config-pmap-p)# drop-connection

hostname(config)# class-map rtsp-traffic-class
hostname(config-cmap)# match default-inspection-traffic

hostname(config)# policy-map rtsp-traffic-policy
hostname(config-pmap)# class rtsp-traffic-class
hostname(config-pmap-c)# inspect rtsp rtsp-filter-map

hostname(config)# service-policy rtsp-traffic-policy global
```

Configure the RTSP Inspection Service Policy

The default ASA configuration includes RTSP inspection on the default port applied globally on all interfaces. A common method for customizing the inspection configuration is to customize the default global policy. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

class-map *name*
match *parameter*

Example:

```
hostname(config)# class-map rtsp_class_map
hostname(config-cmap)# match access-list rtsp
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

policy-map *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for RTSP inspection.

class *name*

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure RTSP inspection.

inspect rtsp [*rtsp_policy_map*]

Where *rtsp_policy_map* is the optional RTSP inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the RTSP inspection policy map, see Configure RTSP Inspection Policy Map, page 14-19.

Example:

```
hostname(config-class)# no inspect rtsp
hostname(config-class)# inspect rtsp rtsp-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different RTSP inspection policy map, you must remove the RTSP inspection with the **no inspect rtsp** command, and then re-add it with the new RTSP inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

SIP Inspection

SIP is a widely used protocol for Internet conferencing, telephony, presence, events notification, and instant messaging. Partially because of its text-based nature and partially because of its flexibility, SIP networks are subject to a large number of security threats.

SIP application inspection provides address translation in message header and body, dynamic opening of ports and basic sanity checks. It also supports application security and protocol conformance, which enforce the sanity of the SIP messages, as well as detect SIP-based attacks.

SIP inspection is enabled by default. You need to configure it only if you want non-default processing, or if you want to identify a TLS proxy to enable encrypted traffic inspection. The following topics explain SIP inspection in more detail.

SIP Inspection Overview

SIP, as defined by the IETF, enables call handling sessions, particularly two-party audio conferences, or "calls." SIP works with SDP for call signaling. SDP specifies the ports for the media stream. Using SIP, the ASA can support any SIP VoIP gateways and VoIP proxy servers. SIP and SDP are defined in the following RFCs:

- SIP: Session Initiation Protocol, RFC 3261
- SDP: Session Description Protocol, RFC 2327

To support SIP calls through the ASA, signaling messages for the media connection addresses, media ports, and embryonic connections for the media must be inspected, because while the signaling is sent over a well-known destination port (UDP/TCP 5060), the media streams are dynamically allocated. Also, SIP embeds IP addresses in the user-data portion of the IP packet. Note that the maximum length of the SIP Request URI that the ASA supports is 255.

Instant Messaging (IM) applications also use SIP extensions (defined in RFC 3428) and SIP-specific event notifications (RFC 3265). After users initiate a chat session (registration/subscription), the IM applications use the MESSAGE/INFO methods and 202 Accept responses when users chat with each other. For example, two users can be online at any time, but not chat for hours. Therefore, the SIP inspection engine opens pinholes that time out according to the configured SIP timeout value. This value must be configured at least five minutes longer than the subscription duration. The subscription duration is defined in the Contact Expires value and is typically 30 minutes.

Because MESSAGE/INFO requests are typically sent using a dynamically allocated port other than port 5060, they are required to go through the SIP inspection engine.

Note SIP inspection supports the Chat feature only. Whiteboard, File Transfer, and Application Sharing are not supported. RTC Client 5.0 is not supported.

Limitations for SIP Inspection

SIP inspection is tested and supported for Cisco Unified Communications Manager (CUCM) 7.0, 8.0, 8.6, and 10.5. It is not supported for CUCM 8.5, or 9.x. SIP inspection might work with other releases and products.

SIP inspection applies NAT for embedded IP addresses. However, if you configure NAT to translate both source and destination addresses, the external address ("from" in the SIP header for the "trying" response message) is not rewritten. Thus, you should use object NAT when working with SIP traffic so that you avoid translating the destination address.

The following limitations and restrictions apply when using PAT with SIP:

- If a remote endpoint tries to register with a SIP proxy on a network protected by the ASA, the registration fails under very specific conditions, as follows:
  - PAT is configured for the remote endpoint.
  - The SIP registrar server is on the outside network.
  - The port is missing in the contact field in the REGISTER message sent by the endpoint to the proxy server.

- If a SIP device transmits a packet in which the SDP portion has an IP address in the owner/creator field (o=) that is different than the IP address in the connection field (c=), the IP address in the o= field may not be properly translated. This is due to a limitation in the SIP protocol, which does not provide a port value in the o= field.
- When using PAT, any SIP header field which contains an internal IP address without a port might not be translated and hence the internal IP address will be leaked outside. If you want to avoid this leakage, configure NAT instead of PAT.

Default SIP Inspection

SIP inspection is enabled by default using the default inspection map, which includes the following:

- SIP instant messaging (IM) extensions: Enabled.
- Non-SIP traffic on SIP port: Permitted.
- Hide server's and endpoint's IP addresses: Disabled.
- Mask software version and non-SIP URIs: Disabled.
- Ensure that the number of hops to destination is greater than 0: Enabled.
- RTP conformance: Not enforced.
- SIP conformance: Do not perform state checking and header validation.

Also note that inspection of encrypted traffic is not enabled. You must configure a TLS proxy to inspect encrypted traffic.

Configure SIP Inspection

SIP application inspection provides address translation in message header and body, dynamic opening of ports and basic sanity checks. It also supports application security and protocol conformance, which enforce the sanity of the SIP messages, as well as detect SIP-based attacks.

SIP inspection is enabled by default. You need to configure it only if you want non-default processing, or if you want to identify a TLS proxy to enable encrypted traffic inspection. If you want to customize SIP inspection, use the following process.

Procedure

Step 1 Configure SIP Inspection Policy Map, page 14-24

Step 2 Configure the SIP Inspection Service Policy, page 14-28

Configure SIP Inspection Policy Map

You can create a SIP inspection policy map to customize SIP inspection actions if the default inspection behavior is not sufficient for your network.

When defining traffic matching criteria, you can either create a class map or include the match statements directly in the policy map. The following procedure explains both approaches.

Before You Begin

Some traffic matching options use regular expressions for matching purposes. If you intend to use one of those techniques, first create the regular expression or regular expression class map.

Procedure

Step 1 (Optional) Create a SIP inspection class map by performing the following steps.

A class map groups multiple traffic matches.You can alternatively identify **match** commands directly in the policy map. The difference between creating a class map and defining the traffic match directly in the inspection policy map is that the class map lets you create more complex match criteria, and you can reuse class maps.

To specify traffic that should not match the class map, use the **match not** command. For example, if the **match not** command specifies the string "example.com," then any traffic that includes "example.com" does not match the class map.

For the traffic that you identify in this class map, you specify actions to take on the traffic in the inspection policy map.

If you want to perform different actions for each **match** command, you should identify the traffic directly in the policy map.

a. Create the class map by entering the following command:

```
hostname(config)# class-map type inspect sip [match-all | match-any] class_map_name
hostname(config-cmap)#
```

Where *the class_map_name* is the name of the class map. The **match-all** keyword is the default, and specifies that traffic must match all criteria to match the class map. The match-any keyword specifies that the traffic matches the class map if it matches at least one **match** statement. The CLI enters class-map configuration mode, where you can enter one or more **match** commands.

b. (Optional) To add a description to the class map, enter the following command:

```
hostname(config-cmap)# description string
```

Where *string* is the description of the class map (up to 200 characters).

c. Specify the traffic on which you want to perform actions using one of the following **match** commands. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

- **match** [**not**] **called-party regex** {*regex_name* | **class** *class_name*}—Matches the called party, as specified in the To header, against the specified regular expression or regular expression class.
- **match** [**not**] **calling-party regex** {*regex_name* | **class** *class_name*}—Matches the calling party, as specified in the From header, against the specified regular expression or regular expression class.
- **match** [**not**] **content length gt** *bytes*—Matches messages where the content length in the SIP header is greater than the specified number of bytes, from 0 to 65536.
- **match** [**not**] **content type** {**sdp** | **regex** {*regex_name* | **class** *class_name*}—Matches the content type as SDP or against the specified regular expression or regular expression class.
- **match** [**not**] **im-subscriber regex** {*regex_name* | **class** *class_name*}—Matches the SIP IM subscriber against the specified regular expression or regular expression class.
- **match** [**not**] **message-path regex** {*regex_name* | **class** *class_name*}—Matches the SIP via header against the specified regular expression or regular expression class.

- **match** [**not**] **request-method** *method*—Matches a SIP request method: ack, bye, cancel, info, invite, message, notify, options, prack, refer, register, subscribe, unknown, update.
- **match** [**not**] **third-party-registration regex** {*regex_name* | **class** *class_name*}—Matches the requester of a third-party registration against the specified regular expression or regular expression class.
- **match** [**not**] **uri {sip | tel} length gt** *bytes*—Matches a URI in the SIP headers of the selected type (SIP or TEL) that is greater than the length specified, between 0 and 65536 bytes.

d. Enter **exit** to leave class map configuration mode.

Step 2 Create a SIP inspection policy map, enter the following command:

```
hostname(config)# policy-map type inspect sip policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 3 (Optional) To add a description to the policy map, enter the following command:

```
hostname(config-pmap)# description string
```

Step 4 To apply actions to matching traffic, perform the following steps.

a. Specify the traffic on which you want to perform actions using one of the following methods:

- If you created a SIP class map, specify it by entering the following command:

```
hostname(config-pmap)# class class_map_name
hostname(config-pmap-c)#
```

- Specify traffic directly in the policy map using one of the **match** commands described for SIP class maps. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

b. Specify the action you want to perform on the matching traffic by entering the following command:

```
hostname(config-pmap-c)# {[drop | drop-connection | reset] [log] |
rate-limit message_rate}
```

Not all options are available for each **match** or **class** command. See the CLI help or the command reference for the exact options available.

The **drop** keyword drops all packets that match.

The **drop-connection** keyword drops the packet and closes the connection.

The **reset** keyword drops the packet, closes the connection, and sends a TCP reset to the server and/or client.

The **log** keyword, which you can use alone or with one of the other keywords, sends a system log message.

The **rate-limit** *message_rate* argument limits the rate of messages. Rate limiting is available for request method matches to “invite” and “register” only.

You can specify multiple **class** or **match** commands in the policy map. For information about the order of **class** and **match** commands, see How Multiple Traffic Classes are Handled, page 12-4.

Step 5 To configure parameters that affect the inspection engine, perform the following steps:

a. To enter parameters configuration mode, enter the following command:

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b. Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- **im**—Enables instant messaging.
- **ip-address-privacy**—Enables IP address privacy, which hides the server and endpoint IP addresses.
- **max-forwards-validation action** {**drop** | **drop-connection** | **reset** | **log**} [**log**]—Checks the value of the Max-Forwards header, which cannot be zero before reaching the destination. You must also choose the action to take for non-conforming traffic (drop packet, drop connection, reset, or log) and whether to enable or disable logging.
- **rtp-conformance** [**enforce-payloadtype**]—Checks RTP packets flowing on the pinholes for protocol conformance. The optional **enforce-payloadtype** keyword enforces the payload type to be audio or video based on the signaling exchange.
- **software-version action** {**mask [log]** | **log**}—Identifies the software version using the Server and User-Agent (endpoint) header fields. You can mask the software version in the SIP messages and optionally log it, or simply log it.
- **state-checking action** {**drop** | **drop-connection** | **reset** | **log**} [**log**]—Enables state transition checking. You must also choose the action to take for non-conforming traffic (drop packet, drop connection, reset, or log) and whether to enable or disable logging.
- **strict-header-validation action** {**drop** | **drop-connection** | **reset** | **log**} [**log**]—Enables strict verification of the header fields in the SIP messages according to RFC 3261. You must also choose the action to take for non-conforming traffic (drop packet, drop connection, reset, or log) and whether to enable or disable logging.
- **traffic-non-sip**—Allows non-SIP traffic on the well-known SIP signaling port.
- **trust-verification-server ip** *ip_address*—Identifies Trust Verification Services servers, which enable Cisco Unified IP Phones to authenticate application servers during HTTPS establishment. You can enter the command up to four times to identify four servers. SIP inspection opens pinholes to each server for each registered phone, and the phone decides which to use. Configure the Trust Verification Services server on the CUCM server.
- **trust-verification-server port** *number*—Identifies the Trust Verification Services port. The default port is 2445, so use this command only if the server uses a different port. The allowed port range is 1026 to 32768.
- **uri-non-sip action** {**mask [log]** | **log**}—Identifies the non-SIP URIs present in the Alert-Info and Call-Info header fields. You can mask the information in the SIP messages and optionally log it, or simply log it.

Example

The following example shows how to disable instant messaging over SIP:

```
hostname(config)# policy-map type inspect sip mymap
hostname(config-pmap)# parameters
hostname(config-pmap-p)# no im

hostname(config)# policy-map global_policy
hostname(config-pmap)# class inspection_default
hostname(config-pmap-c)# inspect sip mymap

hostname(config)# service-policy global_policy global
```

The following example shows how to identify four Trust Verification Services servers.

```
hostname(config)# policy-map type inspect sip sample_sip_map
hostname(config-pmap)# parameters
hostname(config-pmap-p)# trust-verification-server ip 10.1.1.1
hostname(config-pmap-p)# trust-verification-server ip 10.1.1.2
hostname(config-pmap-p)# trust-verification-server ip 10.1.1.3
hostname(config-pmap-p)# trust-verification-server ip 10.1.1.4
hostname(config-pmap-p)# trust-verification-server port 2445
```

Configure the SIP Inspection Service Policy

The default ASA configuration includes SIP inspection on the default port applied globally on all interfaces. A common method for customizing the inspection configuration is to customize the default global policy. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

class-map *name*
match *parameter*

Example:

```
hostname(config)# class-map sip_class_map
hostname(config-cmap)# match access-list sip
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

policy-map *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for SIP inspection.

class *name*

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure SIP inspection.

inspect sip [*sip_policy_map*] [**tls-proxy** *proxy_name*]

Where:

- *sip_policy_map* is the optional SIP inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the SIP inspection policy map, see Configure SIP Inspection Policy Map, page 14-24.
- **tls-proxy** *proxy_name* identifies the TLS proxy to use for this inspection. You need a TLS proxy only if you want to enable inspection of encrypted traffic.

Example:

```
hostname(config-class)# no inspect sip
hostname(config-class)# inspect sip sip-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different SIP inspection policy map, you must remove the SIP inspection with the **no inspect sip** command, and then re-add it with the new SIP inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Configure SIP Timeout Values

The media connections are torn down within two minutes after the connection becomes idle. This is, however, a configurable timeout and can be set for a shorter or longer period of time.

You can configure several SIP global timeout values on the **Configuration > Firewall > Advanced > Global Timeouts** page.

To configure the timeout for the SIP control connection, enter the following command:

```
hostname(config)# timeout sip hh:mm:ss
```

This command configures the idle timeout after which a SIP control connection is closed.

To configure the timeout for the SIP media connection, enter the following command:

```
hostname(config)# timeout sip_media hh:mm:ss
```

This command configures the idle timeout after which a SIP media connection is closed.

Verifying and Monitoring SIP Inspection

The **show sip** command displays information for SIP sessions established across the ASA. Along with the **debug sip** and **show local-host** commands, this command is used for troubleshooting SIP inspection engine issues.

The following is sample output from the **show sip** command:

```
hostname# show sip
Total: 2
call-id c3943000-960ca-2e43-228f@10.130.56.44
    state Call init, idle 0:00:01
call-id c3943000-860ca-7e1f-11f7@10.130.56.45
    state Active, idle 0:00:06
```

This sample shows two active SIP sessions on the ASA (as shown in the Total field). Each call-id represents a call.

The first session, with the call-id c3943000-960ca-2e43-228f@10.130.56.44, is in the state Call Init, which means the session is still in call setup. Call setup is not complete until a final response to the call has been received. For instance, the caller has already sent the INVITE, and maybe received a 100 Response, but has not yet seen the 200 OK, so the call setup is not complete yet. Any non-1xx response message is considered a final response. This session has been idle for 1 second.

The second session is in the state Active, in which call setup is complete and the endpoints are exchanging media. This session has been idle for 6 seconds.

Skinny (SCCP) Inspection

The following sections describe SCCP application inspection.

SCCP Inspection Overview

Skinny (SCCP) is a simplified protocol used in VoIP networks. Cisco IP Phones using SCCP can coexist in an H.323 environment. When used with Cisco CallManager, the SCCP client can interoperate with H.323 compliant terminals.

The ASA supports PAT and NAT for SCCP. PAT is necessary if you have more IP phones than global IP addresses for the IP phones to use. By supporting NAT and PAT of SCCP Signaling packets, Skinny application inspection ensures that all SCCP signaling and media packets can traverse the ASA.

Normal traffic between Cisco CallManager and Cisco IP Phones uses SCCP and is handled by SCCP inspection without any special configuration. The ASA also supports DHCP options 150 and 66, which it accomplishes by sending the location of a TFTP server to Cisco IP Phones and other DHCP clients. Cisco IP Phones might also include DHCP option 3 in their requests, which sets the default route.

Note The ASA supports inspection of traffic from Cisco IP Phones running SCCP protocol version 22 and earlier.

Supporting Cisco IP Phones

In topologies where Cisco CallManager is located on the higher security interface with respect to the Cisco IP Phones, if NAT is required for the Cisco CallManager IP address, the mapping must be **static** as a Cisco IP Phone requires the Cisco CallManager IP address to be specified explicitly in its configuration. An static identity entry allows the Cisco CallManager on the higher security interface to accept registrations from the Cisco IP Phones.

Cisco IP Phones require access to a TFTP server to download the configuration information they need to connect to the Cisco CallManager server.

When the Cisco IP Phones are on a lower security interface compared to the TFTP server, you must use an ACL to connect to the protected TFTP server on UDP port 69. While you do need a static entry for the TFTP server, this does not have to be an identity static entry. When using NAT, an identity static entry maps to the same IP address. When using PAT, it maps to the same IP address and port.

When the Cisco IP Phones are on a *higher* security interface compared to the TFTP server and Cisco CallManager, no ACL or static entry is required to allow the Cisco IP Phones to initiate the connection.

Limitations for SCCP Inspection

SCCP inspection is tested and supported for Cisco Unified Communications Manager (CUCM) 7.0, 8.0, 8.6, and 10.5. It is not supported for CUCM 8.5, or 9.x. SCCP inspection might work with other releases and products.

If the address of an internal Cisco CallManager is configured for NAT or PAT to a different IP address or port, registrations for external Cisco IP Phones fail because the ASA currently does not support NAT or PAT for the file content transferred over TFTP. Although the ASA supports NAT of TFTP messages and opens a pinhole for the TFTP file, the ASA cannot translate the Cisco CallManager IP address and port embedded in the Cisco IP Phone configuration files that are transferred by TFTP during phone registration.

Note The ASA supports stateful failover of SCCP calls except for calls that are in the middle of call setup.

Default SCCP Inspection

SCCP inspection is enabled by default using these defaults:

- Registration: Not enforced.
- Maximum message ID: 0x181.
- Minimum prefix length: 4
- Media timeout: 00:05:00
- Signaling timeout: 01:00:00.
- RTP conformance: Not enforced.

Also note that inspection of encrypted traffic is not enabled. You must configure a TLS proxy to inspect encrypted traffic.

Configure SCCP (Skinny) Inspection

SCCP (Skinny) application inspection performs translation of embedded IP address and port numbers within the packet data, and dynamic opening of pinholes. It also performs additional protocol conformance checks and basic state tracking.

SCCP inspection is enabled by default. You need to configure it only if you want non-default processing, or if you want to identify a TLS proxy to enable encrypted traffic inspection. If you want to customize SCCP inspection, use the following process.

Procedure

Step 1 Configure a Skinny (SCCP) Inspection Policy Map for Additional Inspection Control, page 14-32.

Step 2 Configure the SCCP Inspection Service Policy, page 14-33.

Configure a Skinny (SCCP) Inspection Policy Map for Additional Inspection Control

To specify actions when a message violates a parameter, create an SCCP inspection policy map. You can then apply the inspection policy map when you enable SCCP inspection.

Procedure

Step 1 Create an SCCP inspection policy map.

```
hostname(config)# policy-map type inspect skinny policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 2 (Optional) Add a description to the policy map.

```
hostname(config-pmap)# description string
```

Step 3 (Optional) Drop traffic based on the station message ID field in SCCP messages.

a. Identify the traffic based on the station message ID value in hexadecimal, from 0x0 to 0xffff. You can either specify a single ID, or a range of IDs, using the **match** [**not**] **message-id** command. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

```
hostname(config-pmap)# match message-id value
hostname(config-pmap)# match message-id range start_value end_value
```

Example:

```
hostname(config-pmap)# match message-id 0x181

hostname(config-pmap)# match message-id range 0x200 0xffff
```

b. Specify the action to perform on matching packets. You can drop the packet and optionally log it.

```
hostname(config-pmap)# drop [log]
```

c. Repeat the process until you identify all message IDs that you want to drop.

Step 4 Configure parameters that affect the inspection engine.

a. Enter parameters configuration mode.

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b. Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- **enforce-registration**—Enforces registration before calls can be placed.
- **message-ID max** *hex_value*—Sets the maximum SCCP station message ID allowed. The message ID is in hex, and the default maximum is 0x181.
- **rtp-conformance** [**enforce-payloadtype**]—Checks RTP packets flowing on the pinholes for protocol conformance. The optional **enforce-payloadtype** keyword enforces the payload type to be audio or video based on the signaling exchange.
- **sccp-prefix-len** {**max** | **min**} *length*—Sets the maximum or minimum SCCP prefix length value allowed. Enter the command twice to set both a minimum and maximum value. The default minimum is 4, there is no default maximum.
- **timeout** {**media** | **signaling**} *time*—Sets the timeouts for media and signaling connections (in hh:mm:ss format). To have no timeout, specify 0 for the number. The default media timeout is 5 minutes, the default signaling timeout is one hour.

Example

The following example shows how to define an SCCP inspection policy map.

```
hostname(config)# policy-map type inspect skinny skinny-map
hostname(config-pmap)# parameters
hostname(config-pmap-p)# enforce-registration
hostname(config-pmap-p)# match message-id range 200 300
hostname(config-pmap-p)# drop log
hostname(config)# class-map inspection_default
hostname(config-cmap)# match default-inspection-traffic
hostname(config)# policy-map global_policy
hostname(config-pmap)# class inspection_default
hostname(config-pmap-c)# inspect skinny skinny-map
hostname(config)# service-policy global_policy global
```

Configure the SCCP Inspection Service Policy

The default ASA configuration includes SCCP inspection on the default port applied globally on all interfaces. A common method for customizing the inspection configuration is to customize the default global policy. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

```
class-map name
match parameter
```

Example:

```
hostname(config)# class-map sccp_class_map
hostname(config-cmap)# match access-list sccp
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

policy-map *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for SCCP inspection.

class *name*

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure SCCP inspection.

inspect skinny [*sccp_policy_map*] [**tls-proxy** *proxy_name*]

Where:

- *sccp_policy_map* is the optional SCCP inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the SCCP inspection policy map, see Configure a Skinny (SCCP) Inspection Policy Map for Additional Inspection Control, page 14-32.
- **tls-proxy** *proxy_name* identifies the TLS proxy to use for this inspection. You need a TLS proxy only if you want to enable inspection of encrypted traffic.

Example:

```
hostname(config-class)# no inspect skinny
hostname(config-class)# inspect skinny sccp-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different SCCP inspection policy map, you must remove the SCCP inspection with the **no inspect skinny** command, and then re-add it with the new SCCP inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Verifying and Monitoring SCCP Inspection

The **show skinny** command assists in troubleshooting SCCP (Skinny) inspection engine issues. The following is sample output from the **show skinny** command under the following conditions. There are two active Skinny sessions set up across the ASA. The first one is established between an internal Cisco IP Phone at local address 10.0.0.11 and an external Cisco CallManager at 172.18.1.33. TCP port 2000 is the CallManager. The second one is established between another internal Cisco IP Phone at local address 10.0.0.22 and the same Cisco CallManager.

```
hostname# show skinny
        LOCAL                   FOREIGN                 STATE
-----------------------------------------------------------------
1       10.0.0.11/52238         172.18.1.33/2000                1
  MEDIA 10.0.0.11/22948         172.18.1.22/20798
2       10.0.0.22/52232         172.18.1.33/2000                1
  MEDIA 10.0.0.22/20798         172.18.1.11/22948
```

The output indicates that a call has been established between two internal Cisco IP Phones. The RTP listening ports of the first and second phones are UDP 22948 and 20798 respectively.

The following is sample output from the **show xlate debug** command for these Skinny connections:

```
hostname# show xlate debug
2 in use, 2 most used
Flags:  D - DNS, d - dump, I - identity, i - inside, n - no random,
        r - portmap, s - static
NAT from inside:10.0.0.11 to outside:172.18.1.11 flags si idle 0:00:16 timeout 0:05:00
NAT from inside:10.0.0.22 to outside:172.18.1.22 flags si idle 0:00:14 timeout 0:05:00
```

History for Voice and Video Protocol Inspection

Feature Name	Releases	Feature Information
SIP, SCCP, and TLS Proxy support for IPv6	9.3(1)	You can now inspect IPv6 traffic when using SIP, SCCP, and TLS Proxy (using SIP or SCCP). We did not modify any commands.
SIP support for Trust Verification Services, NAT66, CUCM 10.5, and model 8831 phones.	9.3(2)	You can now configure Trust Verification Services servers in SIP inspection. You can also use NAT66. SIP inspection has been tested with CUCM 10.5. We added the **trust-verification-server** parameter command.

Feature Name	Releases	Feature Information
Improved SIP inspection performance on multiple core ASA.	9.4(1)	If you have multiple SIP signaling flows going through an ASA with multiple cores, SIP inspection performance has been improved. However, you will not see improved performance if you are using a TLS, phone, or IME proxy. We did not modify any commands.
SIP inspection support in ASA clustering	9.4(1)	You can now configure SIP inspection on the ASA cluster. A control flow can be created on any unit (due to load balancing), but its child data flows must reside on the same unit. TLS Proxy configuration is not supported. We introduced the following command: **show cluster service-policy**.
SIP inspection support for Phone Proxy and UC-IME Proxy was removed.	9.4(1)	You can no longer use Phone Proxy or UC-IME Proxy when configuring SIP inspection. Use TLS Proxy to inspect encrypted traffic. We removed the following commands: **phone-proxy**, **uc-ime**. We removed the **phone-proxy** and **uc-ime** keywords from the **inspect sip** command.

Inspection of Database, Directory, and Management Protocols

The following topics explain application inspection for database, directory, and management protocols. For information on why you need to use inspection for certain protocols, and the overall methods for applying inspection, see Getting Started with Application Layer Protocol Inspection, page 12-1.

DCERPC Inspection

The following sections describe the DCERPC inspection engine.

DCERPC Overview

DCERPC is a protocol widely used by Microsoft distributed client and server applications that allows software clients to execute programs on a server remotely.

This typically involves a client querying a server called the Endpoint Mapper listening on a well known port number for the dynamically allocated network information of a required service. The client then sets up a secondary connection to the server instance providing the service. The security appliance allows the appropriate port number and network address and also applies NAT, if needed, for the secondary connection.

DCERPC inspection maps inspect for native TCP communication between the EPM and client on well known TCP port 135. Map and lookup operations of the EPM are supported for clients. Client and server can be located in any security zone. The embedded server IP address and Port number are received from the applicable EPM response messages. Since a client may attempt multiple connections to the server port returned by EPM, multiple use of pinholes are allowed, which have configurable timeouts.

DCE inspection supports the following UUIDs and messages:

- End point mapper (EPM) UUID. All EPM messages are supported.
- ISystemMapper UUID (non-EPM). Supported messages are:
  - RemoteCreateInstance opnum4
  - RemoteGetClassObject opnum3
- Any message that does not contain an IP address or port information because these messages do not require inspection.

Configure DCERPC Inspection

DCERPC inspection is not enabled by default. You must configure it if you want DCERPC inspection.

Procedure

Step 1 Configure a DCERPC Inspection Policy Map, page 15-2.

Step 2 Configure the DCERPC Inspection Service Policy, page 15-3.

Configure a DCERPC Inspection Policy Map

To specify additional DCERPC inspection parameters, create a DCERPC inspection policy map. You can then apply the inspection policy map when you enable DCERPC inspection.

Before You Begin

Some traffic matching options use regular expressions for matching purposes. If you intend to use one of those techniques, first create the regular expression or regular expression class map.

Procedure

Step 1 Create a DCERPC inspection policy map, enter the following command:

```
hostname(config)# policy-map type inspect dcerpc policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 2 (Optional) To add a description to the policy map, enter the following command:

```
hostname(config-pmap)# description string
```

Step 3 To configure parameters that affect the inspection engine, perform the following steps:

a. To enter parameters configuration mode, enter the following command:

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b. Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- **timeout pinhole** *hh:mm:ss*—Configures the timeout for DCERPC pinholes and override the global system pinhole timeout of two minutes. The timeout can be from 00:00:01 to 119:00:00.
- **endpoint-mapper** [**epm-service-only**] [**lookup-operation** [**timeout** *hh:mm:ss*]]—Configures options for the endpoint mapper traffic. The **epm-service-only** keyword enforces endpoint mapper service during binding so that only its service traffic is processed. The **lookup-operation** keyword enables the lookup operation of the endpoint mapper service. You can configure the timeout for pinholes generated from the lookup operation. If no timeout is configured for the lookup operation, the timeout pinhole command or the default is used.

Example

The following example shows how to define a DCERPC inspection policy map with the timeout configured for DCERPC pinholes.

```
hostname(config)# policy-map type inspect dcerpc dcerpc_map
hostname(config-pmap)# timeout pinhole 0:10:00

hostname(config)# class-map dcerpc
hostname(config-cmap)# match port tcp eq 135

hostname(config)# policy-map global-policy
hostname(config-pmap)# class dcerpc
hostname(config-pmap-c)# inspect dcerpc dcerpc-map

hostname(config)# service-policy global-policy global
```

Configure the DCERPC Inspection Service Policy

DCERPC inspection is not enabled in the default inspection policy, so you must enable it if you need this inspection. You can simply edit the default global inspection policy to add DCERPC inspection. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

```
class-map name
match parameter
```

Example:

```
hostname(config)# class-map dcerpc_class_map
hostname(config-cmap)# match access-list dcerpc
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

policy-map *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for DCERPC inspection.

class *name*

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure DCERPC inspection.

inspect dcerpc [*dcerpc_policy_map*]

Where *dcerpc_policy_map* is the optional DCERPC inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the inspection policy map, see Configure a DCERPC Inspection Policy Map, page 15-2.

Example:

```
hostname(config-class)# no inspect dcerpc
hostname(config-class)# inspect dcerpc dcerpc-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different inspection policy map, you must remove the DCERPC inspection with the **no inspect dcerpc** command, and then re-add it with the new inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

GTP Inspection

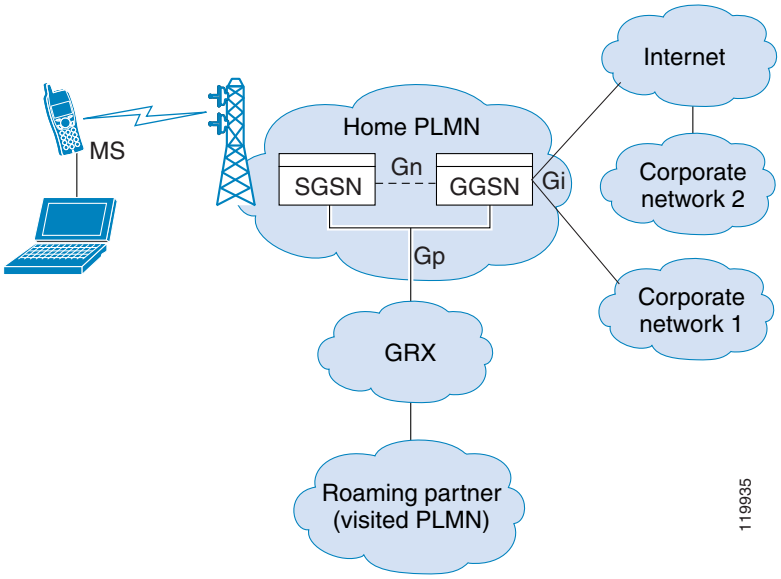
The following sections describe the GTP inspection engine.

Note GTP inspection requires a special license.

GTP Inspection Overview

GPRS provides uninterrupted connectivity for mobile subscribers between GSM networks and corporate networks or the Internet. The GGSN is the interface between the GPRS wireless data network and other networks. The SGSN performs mobility, data session management, and data compression.

Figure 15-1 GPRS Tunneling Protocol



The UMTS is the commercial convergence of fixed-line telephony, mobile, Internet and computer technology. UTRAN is the networking protocol used for implementing wireless networks in this system. GTP allows multi-protocol packets to be tunneled through a UMTS/GPRS backbone between a GGSN, an SGSN and the UTRAN.

GTP does not include any inherent security or encryption of user data, but using GTP with the ASA helps protect your network against these risks.

The SGSN is logically connected to a GGSN using GTP. GTP allows multiprotocol packets to be tunneled through the GPRS backbone between GSNs. GTP provides a tunnel control and management protocol that allows the SGSN to provide GPRS network access for a mobile station by creating, modifying, and deleting tunnels. GTP uses a tunneling mechanism to provide a service for carrying user data packets.

Note When using GTP with failover, if a GTP connection is established and the active unit fails before data is transmitted over the tunnel, the GTP data connection (with a "j" flag set) is not replicated to the standby unit. This occurs because the active unit does not replicate embryonic connections to the standby unit.

Defaults for GTP Inspection

GTP inspection is not enabled by default. However, if you enable it without specifying your own inspection map, a default map is used which provides the following processing. You need to configure a map only if you want different values.

- Errors are not permitted.
- The maximum number of requests is 200.
- The maximum number of tunnels is 500.
- The GSN timeout is 30 minutes.
- The PDP context timeout is 30 minutes.
- The request timeout is 1 minute.
- The signaling timeout is 30 minutes.
- The tunneling timeout is 1 hour.
- The T3 response timeout is 20 seconds.
- Unknown message IDs are dropped and logged.

Configure GTP Inspection

GTP inspection is not enabled by default. You must configure it if you want GTP inspection.

Procedure

Step 1 Configure a GTP Inspection Policy Map, page 15-7.

Step 2 Configure the GTP Inspection Service Policy, page 15-9.

Step 3 (Optional) Configure RADIUS accounting inspection to protect against over-billing attacks. See ILS Inspection, page 15-12.

Configure a GTP Inspection Policy Map

If you want to enforce additional parameters on GTP traffic, and the default map does not meet your needs, create and configure a GTP map.

Before You Begin

Some traffic matching options use regular expressions for matching purposes. If you intend to use one of those techniques, first create the regular expression or regular expression class map.

Procedure

Step 1 Create a GTP inspection policy map:

```
hostname(config)# policy-map type inspect gtp policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 2 (Optional) To add a description to the policy map, enter the following command:

```
hostname(config-pmap)# description string
```

Step 3 To apply actions to matching traffic, perform the following steps.

a. Specify the traffic on which you want to perform actions using one of the following **match** commands. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

- **match** [**not**] **apn regex** {*regex_name* | **class** *class_name*}—Matches the access point name (APN) against the specified regular expression or regular expression class.
- **match** [**not**] **message id** {*message_id* | **range** *message_id_1 message_id_2*}—Matches the message ID, which can be 1 to 255. You can specify a single ID or a range of IDs.
- **match** [**not**] **message length min** *bytes* **max** *bytes*—Matches messages where the length of the UDP payload (GTP header plus the rest of the message) is between the minimum and maximum values, from 1 to 65536.
- **match** [**not**] **version** {*version_id* | **range** *version_id_1 version_id_2*}—Matches the GTP version, which can be 0 to 255. You can specify a single version or a range of versions.

b. Specify the action you want to perform on the matching traffic by entering the following command:

```
hostname(config-pmap-c)# {drop [log]| log | rate-limit message_rate}
```

Not all options are available for each **match** command.

- The **drop** keyword drops the packet.
- The **log** keyword, which you can use alone or with **drop**, sends a system log message.
- The **rate-limit** *message_rate* argument limits the rate of messages. This option is available with **message id** only.

You can specify multiple **match** commands in the policy map. For information about the order of **match** commands, see How Multiple Traffic Classes are Handled, page 12-4.

Step 4 To configure parameters that affect the inspection engine, perform the following steps:

a. To enter parameters configuration mode, enter the following command:

```
hostname(config-pmap)# parameters
```

```
hostname(config-pmap-p)#
```

b. Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- **permit errors**—Allows invalid GTP packets or packets that otherwise would fail parsing and be dropped.
- **request-queue** *max_requests*—Sets the maximum number of GTP requests that will be queued waiting for a response. The default is 200. When the limit has been reached and a new request arrives, the request that has been in the queue for the longest time is removed. The Error Indication, the Version Not Supported and the SGSN Context Acknowledge messages are not considered as requests and do not enter the request queue to wait for a response.
- **tunnel-limit** *max_tunnels*—Sets the maximum number of GTP tunnels allowed to be active on the ASA. The default is 500. New requests will be dropped once the number of tunnels specified by this command is reached.
- **timeout** {**gsn** | **pdp-context** | **request** | **signaling** | **tunnel**} *time*—Sets the idle timeout for the specified service (in hh:mm:ss format). To have no timeout, specify 0 for the number. Enter the command separately for each timeout.

  The **gsn** keyword specifies the period of inactivity after which a GSN will be removed.

  The **pdp-context** keyword specifies the maximum period of time allowed before beginning to receive the PDP context.

  The **request** keyword specifies the maximum period of time allowed before beginning to receive the GTP message.

  The **signaling** keyword specifies the period of inactivity after which the GTP signaling will be removed.

  The **tunnel** keyword specifies the period of inactivity after which the GTP tunnel will be torn down.

Step 5 While still in parameter configuration mode, configure IMSI prefix filtering, if desired.

```
hostname(config-pmap-p)# mcc country_code mnc network_code
```

By default, the security appliance does not check for valid Mobile Country Code (MCC)/Mobile Network Code (MNC) combinations. If you configure IMSI prefix filtering, the MCC and MNC in the IMSI of the received packet is compared with the configured MCC/MNC combinations and is dropped if it does not match.

The Mobile Country Code is a non-zero, three-digit value; add zeros as a prefix for one- or two-digit values. The Mobile Network Code is a two- or three-digit value.

Add all permitted MCC and MNC combinations. By default, the ASA does not check the validity of MNC and MCC combinations, so you must verify the validity of the combinations configured. To find more information about MCC and MNC codes, see the ITU E.212 recommendation, *Identification Plan for Land Mobile Stations*.

Step 6 While still in parameter configuration mode, configure GSN pooling, if desired.

```
hostname(config-pmap-p)# permit response to-object-group SGSN_name
from-object-group GSN_pool
```

When the ASA performs GTP inspection, by default the ASA drops GTP responses from GSNs that were not specified in the GTP request. This situation occurs when you use load-balancing among a pool of GSNs to provide efficiency and scalability of GPRS.

To configure GSN pooling and thus support load balancing, create a network object group that specifies the GSNs and specify this on the **from-object-group** parameter. Likewise, create a network object group for the SGSN and select it as on the **to-object-group** parameter. If the GSN responding belongs to the same object group as the GSN that the GTP request was sent to and if the SGSN is in an object group that the responding GSN is permitted to send a GTP response to, the ASA permits the response.

The network object group can identify the GSN or SGSN by host address or by the subnet that contains them.

Example

The following example shows how to support GSN pooling by defining network objects for the GSN pool and the SGSN. An entire Class C network is defined as the GSN pool but you can identify multiple individual IP addresses, one per **network-object** command, instead of identifying whole networks. The example then modifies a GTP inspection map to permit responses from the GSN pool to the SGSN.

```
hostname(config)# object-group network gsnpool32
hostname(config-network)# network-object 192.168.100.0 255.255.255.0
hostname(config)# object-group network sgsn32
hostname(config-network)# network-object host 192.168.50.100

hostname(config)# policy-map type inspect gtp gtp-policy
hostname(config)# gtp-map gtp-policy
hostname(config-pmap)# parameters
hostname(config-pmap-p)# permit response to-object-group sgsn32
from-object-group gsnpool32
```

Example

The following example shows how to limit the number of tunnels in the network:

```
hostname(config)# policy-map type inspect gtp gmap
hostname(config-pmap)# parameters
hostname(config-pmap-p)# tunnel-limit 3000

hostname(config)# policy-map global_policy
hostname(config-pmap)# class inspection_default
hostname(config-pmap-c)# inspect gtp gmap

hostname(config)# service-policy global_policy global
```

Configure the GTP Inspection Service Policy

GTP inspection is not enabled in the default inspection policy, so you must enable it if you need this inspection. You can simply edit the default global inspection policy to add GTP inspection. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

```
class-map name
match parameter
```

Example:

```
hostname(config)# class-map gtp_class_map
hostname(config-cmap)# match access-list gtp
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

policy-map *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for GTP inspection.

class *name*

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Configure GTP inspection.

inspect gtp [*gtp_policy_map*]

Where *gtp_policy_map* is the optional GTP inspection policy map. You need a map only if you want non-default inspection processing. For information on creating the inspection policy map, see Configure a GTP Inspection Policy Map, page 15-7.

Example:

```
hostname(config-class)# no inspect gtp
hostname(config-class)# inspect gtp gtp-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different inspection policy map, you must remove the GTP inspection with the **no inspect gtp** command, and then re-add it with the new inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Verifying and Monitoring GTP Inspection

To display the GTP configuration, enter the **show service-policy inspect gtp** command in privileged EXEC mode.

Use the **show service-policy inspect gtp statistics** command to show the statistics for GTP inspection. The following is sample output from the **show service-policy inspect gtp statistics** command:

```
hostname# show service-policy inspect gtp statistics
GPRS GTP Statistics:
  version_not_support               0     msg_too_short             0
  unknown_msg                       0     unexpected_sig_msg        0
  unexpected_data_msg               0     ie_duplicated             0
  mandatory_ie_missing              0     mandatory_ie_incorrect    0
  optional_ie_incorrect             0     ie_unknown                0
  ie_out_of_order                   0     ie_unexpected             0
  total_forwarded                   0     total_dropped             0
  signalling_msg_dropped            0     data_msg_dropped          0
  signalling_msg_forwarded          0     data_msg_forwarded        0
  total created_pdp                 0     total deleted_pdp         0
  total created_pdpmcb              0     total deleted_pdpmcb      0
  pdp_non_existent                  0
```

The following is sample GSN output from the **show service-policy inspect gtp statistics gsn** command:

```
hostname# show service-policy inspect gtp statistics gsn 10.9.9.9
1 in use, 1 most used, timeout 0:00:00

GTP GSN Statistics for 10.9.9.9, Idle 0:00:00, restart counter 0
    Tunnels Active 0Tunnels Created 0
    Tunnels Destroyed 0
    Total Messages Received 2
    Signaling Messages Data Messages
    total received 2 0
    dropped 0 0
    forwarded 2 0
```

Use the **show service-policy inspect gtp pdp-context** command to display PDP context-related information. For example:

```
hostname# show service-policy inspect gtp pdp-context detail
1 in use, 1 most used, timeout 0:00:00

Version TID                 MS Addr        SGSN Addr    Idle      APN
v1      1234567890123425         10.0.1.1        10.0.0.2 0:00:13  gprs.cisco.com

    user_name (IMSI): 214365870921435    MS address:         1.1.1.1
    primary pdp: Y                       nsapi: 2
    sgsn_addr_signal:        10.0.0.2    sgsn_addr_data:          10.0.0.2
    ggsn_addr_signal:        10.1.1.1    ggsn_addr_data:          10.1.1.1
    sgsn control teid:     0x000001d1    sgsn data teid:        0x000001d3
    ggsn control teid:     0x6306ffa0    ggsn data teid:        0x6305f9fc
    seq_tpdu_up:                  0    seq_tpdu_down:                 0
    signal_sequence:              0
    upstream_signal_flow:         0     upstream_data_flow:          0
    downstream_signal_flow:       0     downstream_data_flow:        0
    RAupdate_flow:                0
```

The PDP context is identified by the tunnel ID, which is a combination of the values for IMSI and NSAPI. A GTP tunnel is defined by two associated PDP contexts in different GSN nodes and is identified with a Tunnel ID. A GTP tunnel is necessary to forward packets between an external packet data network and a MS user.

ILS Inspection

The ILS inspection engine provides NAT support for Microsoft NetMeeting, SiteServer, and Active Directory products that use LDAP to exchange directory information with an ILS server.

The ASA supports NAT for ILS, which is used to register and locate endpoints in the ILS or SiteServer Directory. PAT cannot be supported because only IP addresses are stored by an LDAP database.

For search responses, when the LDAP server is located outside, NAT should be considered to allow internal peers to communicate locally while registered to external LDAP servers. For such search responses, xlates are searched first, and then DNAT entries to obtain the correct address. If both of these searches fail, then the address is not changed. For sites using NAT 0 (no NAT) and not expecting DNAT interaction, we recommend that the inspection engine be turned off to provide better performance.

Additional configuration may be necessary when the ILS server is located inside the ASA border. This would require a hole for outside clients to access the LDAP server on the specified port, typically TCP 389.

Note Because ILS traffic (H225 call signaling) only occurs on the secondary UDP channel, the TCP connection is disconnected after the TCP inactivity interval. By default, this interval is 60 minutes and can be adjusted using the TCP **timeout** command. In ASDM, this is on the **Configuration > Firewall > Advanced > Global Timeouts** pane.

ILS/LDAP follows a client/server model with sessions handled over a single TCP connection. Depending on the client's actions, several of these sessions may be created.

During connection negotiation time, a BIND PDU is sent from the client to the server. Once a successful BIND RESPONSE from the server is received, other operational messages may be exchanged (such as ADD, DEL, SEARCH, or MODIFY) to perform operations on the ILS Directory. The ADD REQUEST and SEARCH RESPONSE PDUs may contain IP addresses of NetMeeting peers, used by H.323 (SETUP and CONNECT messages) to establish the NetMeeting sessions. Microsoft NetMeeting v2.X and v3.X provides ILS support.

The ILS inspection performs the following operations:

- Decodes the LDAP REQUEST/RESPONSE PDUs using the BER decode functions.
- Parses the LDAP packet.
- Extracts IP addresses.
- Translates IP addresses as necessary.
- Encodes the PDU with translated addresses using BER encode functions.
- Copies the newly encoded PDU back to the TCP packet.
- Performs incremental TCP checksum and sequence number adjustment.

ILS inspection has the following limitations:

- Referral requests and responses are not supported.
- Users in multiple directories are not unified.
- Single users having multiple identities in multiple directories cannot be recognized by NAT.

For information on enabling ILS inspection, see Configure Application Layer Protocol Inspection, page 12-9.

RADIUS Accounting Inspection

The following sections describe the RADIUS Accounting inspection engine.

RADIUS Accounting Inspection Overview

The purpose of RADIUS accounting inspection is to prevent over-billing attacks on GPRS networks that use RADIUS servers. Although you do not need the GTP/GPRS license to implement RADIUS accounting inspection, it has no purpose unless you are implementing GTP inspection and you have a GPRS setup.

The over-billing attack in GPRS networks results in consumers being billed for services that they have not used. In this case, a malicious attacker sets up a connection to a server and obtains an IP address from the SGSN. When the attacker ends the call, the malicious server will still send packets to it, which gets dropped by the GGSN, but the connection from the server remains active. The IP address assigned to the malicious attacker gets released and reassigned to a legitimate user who will then get billed for services that the attacker will use.

RADIUS accounting inspection prevents this type of attack by ensuring the traffic seen by the GGSN is legitimate. With the RADIUS accounting feature properly configured, the ASA tears down a connection based on matching the Framed IP attribute in the Radius Accounting Request Start message with the Radius Accounting Request Stop message. When the Stop message is seen with the matching IP address in the Framed IP attribute, the ASA looks for all connections with the source matching the IP address.

You have the option to configure a secret pre-shared key with the RADIUS server so the ASA can validate the message. If the shared secret is not configured, the ASA will only check that the source IP address is one of the configured addresses allowed to send the RADIUS messages.

Note When using RADIUS accounting inspection with GPRS enabled, the ASA checks for the 3GPP-Session-Stop-Indicator in the Accounting Request STOP messages to properly handle secondary PDP contexts. Specifically, the ASA requires that the Accounting Request STOP messages include the 3GPP-SGSN-Address attribute before it will terminate the user sessions and all associated connections. Some third-party GGSNs might not send this attribute by default.

Configure RADIUS Accounting Inspection

RADIUS accounting inspection is not enabled by default. You must configure it if you want RADIUS accounting inspection.

Procedure

Step 1 Configure a RADIUS Accounting Inspection Policy Map, page 15-14.

Step 2 Configure the RADIUS Accounting Inspection Service Policy, page 15-15.

Configure a RADIUS Accounting Inspection Policy Map

You must create a RADIUS accounting inspection policy map to configure the attributes needed for the inspection.

Procedure

Step 1 Create a RADIUS accounting inspection policy map:

```
hostname(config)# policy-map type inspect radius-accounting policy_map_name
hostname(config-pmap)#
```

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 2 (Optional) Add a description to the policy map.

```
hostname(config-pmap)# description string
```

Step 3 Enter parameters configuration mode.

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

Step 4 Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option.

- **send response**—Instructs the ASA to send Accounting-Request Start and Stop messages to the sender of those messages (which are identified in the **host** command).
- **enable gprs**—Implement GPRS over-billing protection. The ASA checks for the 3GPP VSA 26-10415 attribute in the Accounting-Request Stop and Disconnect messages in order to properly handle secondary PDP contexts. If this attribute is present, then the ASA tears down all connections that have a source IP matching the User IP address on the configured interface.
- **validate-attribute** *number*—Additional criteria to use when building a table of user accounts when receiving Accounting-Request Start messages. These attributes help when the ASA decides whether to tear down connections.

  If you do not specify additional attributes to validate, the decision is based solely on the IP address in the Framed IP Address attribute. If you configure additional attributes, and the ASA receives a start accounting message that includes an address that is currently being tracked, but the other attributes to validate are different, then all connections started using the old attributes are torn down, on the assumption that the IP address has been reassigned to a new user.

  Values range from 1-191, and you can enter the command multiple times. For a list of attribute numbers and their descriptions, see http://www.iana.org/assignments/radius-types.
- **host** *ip_address* [**key** *secret*]—The IP address of the RADIUS server or GGSN. You can optionally include a secret key so that the ASA can validate the message. Without the key, only the IP address is checked. You can repeat this command to identify multiple RADIUS and GGSNs hosts. The ASA receives a copy of the RADIUS accounting messages from these hosts.
- **timeout users** *time*—Sets the idle timeout for users (in hh:mm:ss format). To have no timeout, specify 00:00:00. The default is one hour.

Example

```
policy-map type inspect radius-accounting radius-acct-pmap
```

```
  parameters
    send response
    enable gprs
    validate-attribute 31
    host 10.2.2.2 key 123456789
    host 10.1.1.1 key 12345
class-map type management radius-class
  match port udp eq radius-acct
policy-map global_policy
  class radius-class
    inspect radius-accounting radius-acct-pmap
```

Configure the RADIUS Accounting Inspection Service Policy

RADIUS accounting inspection is not enabled in the default inspection policy, so you must enable it if you need this inspection. Because RADIUS accounting inspection is for traffic directed to the ASA, you must configure it as a management inspection rule rather than a standard rule.

Procedure

Step 1 Create an L3/L4 management class map to identify the traffic for which you want to apply the inspection, and identify the matching traffic.

class-map type management *name*
match {port | access-list} *parameter*

Example:

```
hostname(config)# class-map type management radius-class-map
hostname(config-cmap)# match port udp eq radius-acct
```

In this example, the match is for the radius-acct UDP port, which is 1646. You can specify a different port, a range of ports (**match port udp range** *number1 number2*) or use **match access-list** *acl_name* and use an ACL.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic.

policy-map *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 management class map you are using for RADIUS accounting inspection.

class *name*

Example:

```
hostname(config-pmap)# class radius-class-map
```

Step 4 Configure RADIUS accounting inspection.

inspect radius-accounting *radius_accounting_policy_map*

Where *radius_accounting_policy_map* is the RADIUS accounting inspection policy map you created in Configure a RADIUS Accounting Inspection Policy Map, page 15-14.

Example:

```
hostname(config-class)# no inspect radius-accounting
hostname(config-class)# inspect radius-accounting radius-class-map
```

Note If you are editing an in-use policy to use a different inspection policy map, you must remove the RADIUS accounting inspection with the **no inspect radius-accounting** command, and then re-add it with the new inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

RSH Inspection

RSH inspection is enabled by default. The RSH protocol uses a TCP connection from the RSH client to the RSH server on TCP port 514. The client and server negotiate the TCP port number where the client listens for the STDERR output stream. RSH inspection supports NAT of the negotiated port number if necessary.

For information on enabling RSH inspection, see Configure Application Layer Protocol Inspection, page 12-9.

SNMP Inspection

SNMP application inspection lets you restrict SNMP traffic to a specific version of SNMP. Earlier versions of SNMP are less secure; therefore, denying certain SNMP versions may be required by your security policy. The ASA can deny SNMP versions 1, 2, 2c, or 3. You control the versions permitted by creating an SNMP map.

SNMP inspection is not enabled in the default inspection policy, so you must enable it if you need this inspection. You can simply edit the default global inspection policy to add SNMP inspection. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 Create an SNMP map.

Use the **snmp-map** *map_name* command to create the map and enter SNMP map configuration mode, then the **deny version** *version* command to identify the versions to disallow. The version can be 1, 2, 2c, or 3.

Example:

The following example denies SNMP Versions 1 and 2:

```
hostname(config)# snmp-map sample_map
hostname(config-snmp-map)# deny version 1
hostname(config-snmp-map)# deny version 2
```

Step 2 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

```
class-map name
match parameter
```

Example:

```
hostname(config)# class-map snmp_class_map
hostname(config-cmap)# match access-list snmp
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 3 Add or edit a policy map that sets the actions to take with the class map traffic.

```
policy-map name
```

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 4 Identify the L3/L4 class map you are using for SNMP inspection.

```
class name
```

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 5 Configure SNMP inspection.

```
inspect snmp [snmp_map]
```

Where *snmp_map* is the optional SNMP inspection policy map. You need a map only if you want non-default inspection processing.

Example:

```
hostname(config-class)# no inspect snmp
hostname(config-class)# inspect snmp snmp-map
```

Note If you are editing the default global policy (or any in-use policy) to use a different inspection policy map, you must remove the SNMP inspection with the **no inspect snmp** command, and then re-add it with the new inspection policy map name.

Step 6 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

SQL*Net Inspection

SQL*Net inspection is enabled by default.

The SQL*Net protocol consists of different packet types that the ASA handles to make the data stream appear consistent to the Oracle applications on either side of the ASA.

The default port assignment for SQL*Net is 1521. This is the value used by Oracle for SQL*Net, but this value does not agree with IANA port assignments for Structured Query Language (SQL). Use the **class-map** command to apply SQL*Net inspection to a range of port numbers.

Note Disable SQL*Net inspection when SQL data transfer occurs on the same port as the SQL control TCP port 1521. The security appliance acts as a proxy when SQL*Net inspection is enabled and reduces the client window size from 65000 to about 16000 causing data transfer issues.

The ASA translates all addresses and looks in the packets for all embedded ports to open for SQL*Net Version 1.

For SQL*Net Version 2, all DATA or REDIRECT packets that immediately follow REDIRECT packets with a zero data length will be fixed up.

The packets that need fix-up contain embedded host/port addresses in the following format:

```
(ADDRESS=(PROTOCOL=tcp)(DEV=6)(HOST=a.b.c.d)(PORT=a))
```

SQL*Net Version 2 TNSFrame types (Connect, Accept, Refuse, Resend, and Marker) will not be scanned for addresses to NAT nor will inspection open dynamic connections for any embedded ports in the packet.

SQL*Net Version 2 TNSFrames, Redirect, and Data packets will be scanned for ports to open and addresses to NAT, if preceded by a REDIRECT TNSFrame type with a zero data length for the payload. When the Redirect message with data length zero passes through the ASA, a flag will be set in the connection data structure to expect the Data or Redirect message that follows to be translated and ports to be dynamically opened. If one of the TNS frames in the preceding paragraph arrive after the Redirect message, the flag will be reset.

The SQL*Net inspection engine will recalculate the checksum, change IP, TCP lengths, and readjust Sequence Numbers and Acknowledgment Numbers using the delta of the length of the new and old message.

SQL*Net Version 1 is assumed for all other cases. TNSFrame types (Connect, Accept, Refuse, Resend, Marker, Redirect, and Data) and all packets will be scanned for ports and addresses. Addresses will be translated and port connections will be opened.

For information on enabling SQL*Net inspection, see Configure Application Layer Protocol Inspection, page 12-9.

Sun RPC Inspection

This section describes Sun RPC application inspection.

Sun RPC Inspection Overview

The Sun RPC inspection engine enables or disables application inspection for the Sun RPC protocol. Sun RPC is used by NFS and NIS. Sun RPC services can run on any port. When a client attempts to access a Sun RPC service on a server, it must learn the port that service is running on. It does this by querying the port mapper process, usually rpcbind, on the well-known port of 111.

The client sends the Sun RPC program number of the service and the port mapper process responds with the port number of the service. The client sends its Sun RPC queries to the server, specifying the port identified by the port mapper process. When the server replies, the ASA intercepts this packet and opens both embryonic TCP and UDP connections on that port.

Tip Sun RPC inspection is enabled by default. You simply need to manage the Sun RPC server table to identify which services are allowed to traverse the firewall. For information on enabling Sun RPC inspection, see Configure Application Layer Protocol Inspection, page 12-9.

The following limitations apply to Sun RPC inspection:

- NAT or PAT of Sun RPC payload information is not supported.
- Sun RPC inspection supports inbound ACLs only. Sun RPC inspection does not support outbound ACLs because the inspection engine uses dynamic ACLs instead of secondary connections. Dynamic ACLs are always added on the ingress direction and not on egress; therefore, this inspection engine does not support outbound ACLs. To view the dynamic ACLs configured for the ASA, use the **show asp table classify domain permit** command.

Managing Sun RPC Services

Use the Sun RPC services table to control Sun RPC traffic through the ASA based on established Sun RPC sessions. To create entries in the Sun RPC services table, use the **sunrpc-server** command in global configuration mode:

```
hostname(config)# sunrpc-server interface_name ip_address mask service service_type
protocol {tcp | udp} port[-port] timeout hh:mm:ss
```

You can use this command to specify the timeout after which the pinhole that was opened by Sun RPC application inspection will be closed. For example, to create a timeout of 30 minutes to the Sun RPC server with the IP address 192.168.100.2, enter the following command:

```
hostname(config)# sunrpc-server inside 192.168.100.2 255.255.255.255 service 100003
protocol tcp 111 timeout 00:30:00
```

This command specifies that the pinhole that was opened by Sun RPC application inspection will be closed after 30 minutes. In this example, the Sun RPC server is on the inside interface using TCP port 111. You can also specify UDP, a different port number, or a range of ports. To specify a range of ports, separate the starting and ending port numbers in the range with a hyphen (for example, 111-113).

The service type identifies the mapping between a specific service type and the port number used for the service. To determine the service type, which in this example is 100003, use the **sunrpcinfo** command at the UNIX or Linux command line on the Sun RPC server machine.

To clear the Sun RPC configuration, enter the following command.

```
hostname(config)# clear configure sunrpc-server
```

This removes the configuration performed using the **sunrpc-server** command. The **sunrpc-server** command allows pinholes to be created with a specified timeout.

To clear the active Sun RPC services, enter the following command:

```
hostname(config)# clear sunrpc-server active
```

This clears the pinholes that are opened by Sun RPC application inspection for specific services, such as NFS or NIS.

Verifying and Monitoring Sun RPC Inspection

The sample output in this section is for a Sun RPC server with an IP address of 192.168.100.2 on the inside interface and a Sun RPC client with an IP address of 209.168.200.5 on the outside interface.

To view information about the current Sun RPC connections, enter the **show conn** command. The following is sample output from the **show conn** command:

```
hostname# show conn
15 in use, 21 most used
UDP out 209.165.200.5:800 in 192.168.100.2:2049 idle 0:00:04 flags -
UDP out 209.165.200.5:714 in 192.168.100.2:111 idle 0:00:04 flags -
UDP out 209.165.200.5:712 in 192.168.100.2:647 idle 0:00:05 flags -
UDP out 192.168.100.2:0 in 209.165.200.5:714 idle 0:00:05 flags i
hostname(config)#
```

To display the information about the Sun RPC service table configuration, enter the **show running-config sunrpc-server** command. The following is sample output from the **show running-config sunrpc-server** command:

```
hostname(config)# show running-config sunrpc-server
sunrpc-server inside 192.168.100.2 255.255.255.255 service 100003 protocol UDP port 111
timeout 0:30:00
sunrpc-server inside 192.168.100.2 255.255.255.255 service 100005 protocol UDP port 111
timeout 0:30:00
```

This output shows that a timeout interval of 30 minutes is configured on UDP port 111 for the Sun RPC server with the IP address 192.168.100.2 on the inside interface.

To display the pinholes open for Sun RPC services, enter the **show sunrpc-server active** command. The following is sample output from **show sunrpc-server active** command:

```
hostname# show sunrpc-server active
LOCAL FOREIGN SERVICE TIMEOUT
-----------------------------------------------
1 209.165.200.5/0 192.168.100.2/2049 100003 0:30:00
2 209.165.200.5/0 192.168.100.2/2049 100003 0:30:00
3 209.165.200.5/0 192.168.100.2/647 100005 0:30:00
4 209.165.200.5/0 192.168.100.2/650 100005 0:30:00
```

The entry in the LOCAL column shows the IP address of the client or server on the inside interface, while the value in the FOREIGN column shows the IP address of the client or server on the outside interface.

To view information about the Sun RPC services running on a Sun RPC server, enter the **rpcinfo -p** command from the Linux or UNIX server command line. The following is sample output from the **rpcinfo -p** command:

```
sunrpcserver:~ # rpcinfo -p
program vers proto port
100000 2 tcp 111 portmapper
100000 2 udp 111 portmapper
100024 1 udp 632 status
100024 1 tcp 635 status
100003 2 udp 2049 nfs
100003 3 udp 2049 nfs
100003 2 tcp 2049 nfs
100003 3 tcp 2049 nfs
100021 1 udp 32771 nlockmgr
100021 3 udp 32771 nlockmgr
100021 4 udp 32771 nlockmgr
100021 1 tcp 32852 nlockmgr
100021 3 tcp 32852 nlockmgr
100021 4 tcp 32852 nlockmgr
100005 1 udp 647 mountd
100005 1 tcp 650 mountd
100005 2 udp 647 mountd
100005 2 tcp 650 mountd
100005 3 udp 647 mountd
100005 3 tcp 650 mountd
```

In this output, port 647 corresponds to the mountd daemon running over UDP. The mountd process would more commonly be using port 32780. The mountd process running over TCP uses port 650 in this example.

XDMCP Inspection

XDMCP inspection is enabled by default; however, the XDMCP inspection engine is dependent upon proper configuration of the **established** command.

XDMCP is a protocol that uses UDP port 177 to negotiate X sessions, which use TCP when established.

For successful negotiation and start of an XWindows session, the ASA must allow the TCP back connection from the Xhosted computer. To permit the back connection, use the **established** command on the ASA. Once XDMCP negotiates the port to send the display, The **established** command is consulted to verify if this back connection should be permitted.

During the XWindows session, the manager talks to the display Xserver on the well-known port 6000 | n. Each display has a separate connection to the Xserver, as a result of the following terminal setting.

```
setenv DISPLAY Xserver:n
```

where *n* is the display number.

When XDMCP is used, the display is negotiated using IP addresses, which the ASA can NAT if needed. XDCMP inspection does not support PAT.

For information on enabling XDMCP inspection, see Configure Application Layer Protocol Inspection, page 12-9.

VXLAN Inspection

Virtual Extensible Local Area Network (VXLAN) inspection works on VXLAN encapsulated traffic that passes through the ASA. It ensures that the VXLAN header format conforms to standards, dropping any malformed packets. VXLAN inspection is not done on traffic for which the ASA acts as a VXLAN Tunnel End Point (VTEP) or a VXLAN gateway, as those checks are done as a normal part of decapsulating VXLAN packets.

VXLAN packets are UDP, normally on port 4789. This port is part of the default-inspection-traffic class, so you can simply add VXLAN inspection to the inspection_default service policy rule. Alternatively, you can create a class for it using port or ACL matching.

History for Database, Directory, and Management Protocol Inspection

Feature Name	Releases	Feature Information
DCERPC inspection support for ISystemMapper UUID message RemoteGetClassObject opnum3.	9.4(1)	The ASA started supporting non-EPM DCERPC messages in release 8.3, supporting the ISystemMapper UUID message RemoteCreateInstance opnum4. This change extends support to the RemoteGetClassObject opnum3 message. We did not modify any commands.
VXLAN packet inspection	9.4(1)	The ASA can inspect the VXLAN header to enforce compliance with the standard format. We introduced the following command: **inspect vxlan**.

PART 4

Connection Management and Threat Detection

Connection Settings

This chapter describes how to configure connection settings for connections that go through the ASA, or for management connections that go to the ASA.

What Are Connection Settings?

Connection settings comprise a variety of features related to managing traffic connections, such as a TCP flow through the ASA. Some features are named components that you would configure to supply specific services.

Connection settings include the following:

- **Global timeouts for various protocols**—All global timeouts have default values, so you need to change them only if you are experiencing premature connection loss.
- **Connection timeouts per traffic class**—You can override the global timeouts for specific types of traffic using service policies. All traffic class timeouts have default values, so you do not have to set them.
- **Connection limits and TCP Intercept**—By default, there are no limits on how many connections can go through (or to) the ASA. You can set limits on particular traffic classes using service policy rules to protect servers from denial of service (DoS) attacks. Particularly, you can set limits on embryonic connections (those that have not finished the TCP handshake), which protects against SYN flooding attacks. When embryonic limits are exceeded, the TCP Intercept component gets involved to proxy connections and ensure that attacks are throttled.
- **Dead Connection Detection (DCD)**—If you have persistent connections that are valid but often idle, so that they get closed because they exceed idle timeout settings, you can enable Dead Connection Detection to identify idle but valid connections and keep them alive (by resetting their idle timers). Whenever idle times are exceeded, DCD probes both sides of the connection to see if both sides agree the connection is valid. The **show service-policy** command includes counters to show the amount of activity from DCD.

- **TCP sequence randomization**—Each TCP connection has two ISNs: one generated by the client and one generated by the server. By default, the ASA randomizes the ISN of the TCP SYN passing in both the inbound and outbound directions. Randomization prevents an attacker from predicting the next ISN for a new connection and potentially hijacking the new session. You can disable randomization per traffic class if desired.
- **TCP Normalization**—The TCP Normalizer protects against abnormal packets. You can configure how some types of packet abnormalities are handled by traffic class.
- **TCP State Bypass**—You can bypass TCP state checking if you use asymmetrical routing in your network.

Configure Connection Settings

Connection limits, timeouts, TCP Normalization, TCP sequence randomization, and decrementing time-to-live (TTL) have default values that are appropriate for most networks. You need to configure these connection settings only if you have unusual requirements, your network has specific types of configuration, or if you are experiencing unusual connection loss due to premature idle timeouts.

TCP Intercept, TCP State Bypass, and Dead Connection Detection (DCD) are not enabled. You would configure these services on specific traffic classes only, and not as a general service.

The following general procedure covers the gamut of possible connection setting configurations. Pick and choose which to implement based on your needs.

Procedure

Step 1 Configure Global Timeouts, page 16-3. These settings change the default idle timeouts for various protocols for all traffic that passes through the device. If you are having problems with connections being reset due to premature timeouts, first try changing the global timeouts.

Step 2 Protect Servers from a SYN Flood DoS Attack (TCP Intercept), page 16-4. Use this procedure to configure TCP Intercept.

Step 3 Customize Abnormal TCP Packet Handling (TCP Maps, TCP Normalizer), page 16-7, if you want to alter the default TCP Normalization behavior for specific traffic classes.

Step 4 Bypass TCP State Checks for Asynchronous Routing (TCP State Bypass), page 16-10, if you have this type of routing environment.

Step 5 Disable TCP Sequence Randomization, page 16-13, if the default randomization is scrambling data for certain connections.

Step 6 Configure Connection Settings for Specific Traffic Classes (All Services), page 16-14. This is a catch-all procedure for connection settings. These settings can override the global defaults for specific traffic classes using service policy rules. You also use these rules to customize TCP Normalizer, change TCP sequence randomization, decrement time-to-live on packets, and implement TCP Intercept, Dead Connection Detection, or TCP State Bypass.

Configure Global Timeouts

You can set the global idle timeout durations for the connection and translation slots of various protocols. If the slot has not been used for the idle time specified, the resource is returned to the free pool. TCP connection slots are freed approximately 60 seconds after a normal connection close sequence.

Changing the global timeout sets a new default timeout, which in some cases can be overridden for particular traffic flows through service policies.

Procedure

Step 1 Use the **timeout** command to set global timeouts.

```
hostname(config)# timeout feature time
```

All timeout values are in the format *hh*:*mm*:*ss*, with a maximum duration of 1193:0:0. Use the **no timeout** command to reset all timeouts to their default values. If you want to simply reset one timer to the default, enter the **timeout** command for that setting with the default value.

Use **0** for the value to disable a timer.

You can configure the following global timeouts.

- **timeout conn** *hh*:*mm*:*ss*—The idle time after which a connection closes, between 0:5:0 and 1193:0:0. The default is 1 hour (1:0:0).
- **timeout half-closed** *hh*:*mm*:*ss*—The idle time until a TCP half-closed connection closes. The minimum is 5 minutes. The default is 10 minutes.
- **timeout udp** *hh*:*mm*:*ss*—The idle time until a UDP connection closes. This duration must be at least 1 minute. The default is 2 minutes.
- **timeout icmp** *hh*:*mm*:*ss*—The idle time for ICMP, between 0:0:2 and 1193:0:0. The default is 2 seconds (0:0:2).
- **timeout sunrpc** *hh*:*mm*:*ss*—The idle time until a SunRPC slot is freed. This duration must be at least 1 minute. The default is 10 minutes.
- **timeout H323** *hh*:*mm*:*ss*—The idle time after which H.245 (TCP) and H.323 (UDP) media connections close, between 0:0:0 and 1193:0:0. The default is 5 minutes (0:5:0). Because the same connection flag is set on both H.245 and H.323 media connections, the H.245 (TCP) connection shares the idle timeout with the H.323 (RTP and RTCP) media connection.
- **timeout h225** *hh*:*mm*:*ss*—The idle time until an H.225 signaling connection closes. The H.225 default timeout is 1 hour (1:0:0). To close a connection immediately after all calls are cleared, a value of 1 second (0:0:1) is recommended.
- **timeout mgcp** *hh*:*mm*:*ss*—The idle time after which an MGCP media connection is removed, between 0:0:0 and 1193:0:0. The default is 5 minutes (0:5:0)
- **timeout mgcp-pat** *hh*:*mm*:*ss*—The absolute interval after which an MGCP PAT translation is removed, between 0:0:0 and 1193:0:0. The default is 5 minutes (0:5:0). The minimum time is 30 seconds.
- **timeout sip** *hh*:*mm*:*ss*—The idle time until a SIP signaling port connection closes, between 0:5:0 and 1193:0:0. The default is 30 minutes (0:30:0).
- **timeout sip_media** *hh*:*mm*:*ss*—The idle time until an SIP media port connection closes. This duration must be at least 1 minute. The default is 2 minutes. The SIP media timer is used used for SIP RTP/RTCP with SIP UDP media packets, instead of the UDP inactivity timeout.

- **timeout sip-provisional-media** *hh*:*mm*:*ss*—The timeout value for SIP provisional media connections, between 0:1:0 and 0:30:0. The default is 2 minutes.
- **timeout sip-invite** *hh*:*mm*:*ss*—The idle time after which pinholes for PROVISIONAL responses and media xlates will be closed, between 0:1:0 and 00:30:0. The default is 3 minutes (0:3:0).
- **timeout sip-disconnect** *hh*:*mm*:*ss*—The idle time after which a SIP session is deleted if the 200 OK is not received for a CANCEL or a BYE message, between 0:0:1 and 00:10:0. The default is 2 minutes (0:2:0).
- **timeout uauth** *hh*:*mm*:*ss* {**absolute** | **inactivity**}—The duration before the authentication and authorization cache times out and the user has to reauthenticate the next connection, between 0:0:0 and 1193:0:0. The default is 5 minutes (0:5:0). The default timer is **absolute**; you can set the timeout to occur after a period of inactivity by entering the **inactivity** keyword. The uauth duration must be shorter than the xlate duration. Set to 0 to disable caching. Do not use 0 if passive FTP is used for the connection or if the virtual http command is used for web authentication.
- **timeout xlate** *hh*:*mm*:*ss*—The idle time until a translation slot is freed. This duration must be at least 1 minute. The default is 3 hours.
- **timeout tcp-proxy-reassembly** *hh*:*mm*:*ss*—The idle timeout after which buffered packets waiting for reassembly are dropped, between 0:0:10 and 1193:0:0. The default is 1 minute (0:1:0).
- **timeout floating-conn** *hh*:*mm*:*ss*—When multiple static routes exist to a network with different metrics, the ASA uses the one with the best metric at the time of connection creation. If a better route becomes available, then this timeout lets connections be closed so a connection can be reestablished to use the better route. The default is 0 (the connection never times out). To take advantage of this feature, change the timeout to a new value between 0:1:0 and 1193:0:0.
- **timeout pat-xlate** *hh*:*mm*:*ss*—The idle time until a PAT translation slot is freed, between 0:0:30 and 0:5:0. The default is 30 seconds. You may want to increase the timeout if upstream routers reject new connections using a freed PAT port because the previous connection might still be open on the upstream device.

Protect Servers from a SYN Flood DoS Attack (TCP Intercept)

A SYN-flooding denial of service (DoS) attack occurs when an attacker sends a series of SYN packets to a host. These packets usually originate from spoofed IP addresses. The constant flood of SYN packets keeps the server SYN queue full, which prevents it from servicing connection requests from legitimate users.

You can limit the number of embryonic connections to help prevent SYN flooding attacks. An embryonic connection is a connection request that has not finished the necessary handshake between source and destination.

When the embryonic connection threshold of a connection is crossed, the ASA acts as a proxy for the server and generates a SYN-ACK response to the client SYN request using the SYN cookie method (see Wikipedia for details on SYN cookies). When the ASA receives an ACK back from the client, it can then authenticate that the client is real and allow the connection to the server. The component that performs the proxy is called TCP Intercept.

Note Ensure that you set the embryonic connection limit lower than the TCP SYN backlog queue on the server that you want to protect. Otherwise, valid clients can no longer access the server during a SYN attack. To determine reasonable values for embryonic limits, carefully analyze the capacity of the server, the network, and server usage.

The end-to-end process for protecting a server from a SYN flood attack involves setting connection limits, enabling TCP Intercept statistics, and then monitoring the results.

Before You Begin

- Ensure that you set the embryonic connection limit lower than the TCP SYN backlog queue on the server that you want to protect. Otherwise, valid clients can no longer access the server during a SYN attack. To determine reasonable values for embryonic limits, carefully analyze the capacity of the server, the network, and server usage.
- Depending on the number of CPU cores on your ASA model, the maximum concurrent and embryonic connections can exceed the configured numbers due to the way each core manages connections. In the worst case scenario, the ASA allows up to *n*-1 extra connections and embryonic connections, where *n* is the number of cores. For example, if your model has 4 cores, if you configure 6 concurrent connections and 4 embryonic connections, you could have an additional 3 of each type. To determine the number of cores for your model, enter the **show cpu core** command.

Procedure

Step 1 Create an L3/L4 class map to identify the servers you are protecting. Use an access-list match.

```
class-map name
match parameter
```

Example:

```
hostname(config)# access-list servers extended permit tcp any host 10.1.1.5 eq http
hostname(config)# access-list servers extended permit tcp any host 10.1.1.6 eq http
hostname(config)# class-map protected-servers
hostname(config-cmap)# match access-list servers
```

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic, and identify the class map.

```
policy-map name
class name
```

Example:

```
hostname(config)# policy-map global_policy
hostname(config-pmap)# class protected-servers
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name. For the class map, specify the class you created earlier in this procedure.

Step 3 Set the embryonic connection limits.

- **set connection embryonic-conn-max** n—The maximum number of simultaneous embryonic connections allowed, between 0 and 2000000. The default is 0, which allows unlimited connections.

- **set connection per-client-embryonic-max** n—The maximum number of simultaneous embryonic connections allowed per client, between 0 and 2000000. The default is 0, which allows unlimited connections.

Example:

```
hostname(config-pmap-c)# set connection embryonic-conn-max 1000
hostname(config-pmap-c)# set connection per-client-embryonic-max 50
```

Step 4 If you are editing an existing service policy (such as the default global policy called global_policy), you can skip this step. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Step 5 Configure threat detection statistics for attacks intercepted by TCP Intercept.

threat-detection statistics tcp-intercept [**rate-interval** *minutes*] [**burst-rate** *attacks_per_sec*] [**average-rate** *attacks_per_sec*]

Example:

```
hostname(config)# threat-detection statistics tcp-intercept
```

The **rate-interval** keyword sets the size of the history monitoring window, between 1 and 1440 minutes. The default is 30 minutes. During this interval, the ASA samples the number of attacks 30 times.

The **burst-rate** keyword sets the threshold for syslog message generation, between 25 and 2147483647. The default is 400 per second. When the burst rate is exceeded, syslog message 733104 is generated.

The **average-rate** keyword sets the average rate threshold for syslog message generation, between 25 and 2147483647. The default is 200 per second. When the average rate is exceeded, syslog message 733105 is generated.

Step 6 Monitor the results with the following commands:

- **show threat-detection statistics top tcp-intercept** [**all** | **detail**]—View the top 10 protected servers under attack. The **all** keyword shows the history data of all the traced servers. The **detail** keyword shows history sampling data. The ASA samples the number of attacks 30 times during the rate interval, so for the default 30 minute period, statistics are collected every 60 seconds.
- **clear threat-detection statistics tcp-intercept**—Erases TCP Intercept statistics.

Example:

```
hostname(config)# show threat-detection statistics top tcp-intercept
Top 10 protected servers under attack (sorted by average rate)
Monitoring window size: 30 mins    Sampling interval: 30 secs
<Rank> <Server IP:Port> <Interface> <Ave Rate> <Cur Rate> <Total> <Source IP (Last Attack
Time)>
--------------------------------------------------------------------------------
1    10.1.1.5:80 inside 1249 9503 2249245 <various> Last: 10.0.0.3 (0 secs ago)
2    10.1.1.6:80 inside 10 10 6080 10.0.0.200 (0 secs ago)
```

Customize Abnormal TCP Packet Handling (TCP Maps, TCP Normalizer)

The TCP Normalizer identifies abnormal packets that the ASA can act on when they are detected; for example, the ASA can allow, drop, or clear the packets. TCP normalization helps protect the ASA from attacks. TCP normalization is always enabled, but you can customize how some features behave.

The default configuration includes the following settings:

```
no check-retransmission
no checksum-verification
exceed-mss allow
queue-limit 0 timeout 4
reserved-bits allow
syn-data allow
synack-data drop
invalid-ack drop
seq-past-window drop
tcp-options range 6 7 clear
tcp-options range 9 255 clear
tcp-options selective-ack allow
tcp-options timestamp allow
tcp-options window-scale allow
ttl-evasion-protection
urgent-flag clear
window-variation allow-connection
```

To customize the TCP normalizer, first define the settings using a TCP map. Then, you can apply the map to selected traffic classes using service policies.

Procedure

Step 1 Create a TCP map to specify the TCP normalization criteria that you want to look for.

```
hostname(config)# tcp-map tcp-map-name
```

Step 2 Configure the TCP map criteria by entering one or more of the following commands. The defaults are used for any commands you do not enter. Use the **no** form of a command to disable the setting.

- **check-retransmission**—Prevent inconsistent TCP retransmissions. This command is disabled by default.
- **checksum-verification**—Verify the TCP checksum, dropping packets that fail verification. This command is disabled by default.
- **exceed-mss** {**allow** | **drop**}—Allow or drop packets whose data length exceeds the TCP maximum segment size. The default is to allow the packets.
- **invalid-ack** {**allow** | **drop**}—Allow or drop packets with an invalid ACK. The default is to drop the packet, with the exception of WAAS connections, where they are allowed. You might see invalid ACKs in the following instances:
  - In the TCP connection SYN-ACK-received status, if the ACK number of a received TCP packet is not exactly the same as the sequence number of the next TCP packet sending out, it is an invalid ACK.
  - Whenever the ACK number of a received TCP packet is greater than the sequence number of the next TCP packet sending out, it is an invalid ACK.

- **queue-limit** *pkt_num* [**timeout** *seconds*]—Set the maximum number of out-of-order packets that can be buffered and put in order for a TCP connection, between 1 and 250 packets. The default is 0, which means this setting is disabled and the default system queue limit is used depending on the type of traffic:
  - Connections for application inspection (the **inspect** command), IPS (the **ips** command), and TCP check-retransmission (the TCP map **check-retransmission** command) have a queue limit of 3 packets. If the ASA receives a TCP packet with a different window size, then the queue limit is dynamically changed to match the advertised setting.
  - For other TCP connections, out-of-order packets are passed through untouched.

  If you set the **queue-limit** command to be 1 or above, then the number of out-of-order packets allowed for all TCP traffic matches this setting. For example, for application inspection, IPS, and TCP check-retransmission traffic, any advertised settings from TCP packets are ignored in favor of the **queue-limit** setting. For other TCP traffic, out-of-order packets are now buffered and put in order instead of passed through untouched.

  The **timeout** *seconds* argument sets the maximum amount of time that out-of-order packets can remain in the buffer, between 1 and 20 seconds; if they are not put in order and passed on within the timeout period, then they are dropped. The default is 4 seconds. You cannot change the timeout for any traffic if the *pkt_num* argument is set to 0; you need to set the limit to be 1 or above for the **timeout** keyword to take effect.
- **reserved-bits** {**allow** | **clear** | **drop**}—Set the action for reserved bits in the TCP header. You can **allow** the packet (without changing the bits), **clear** the bits and allow the packet, or **drop** the packet.
- **seq-past-window** {**allow** | **drop**}—Set the action for packets that have past-window sequence numbers, namely the sequence number of a received TCP packet is greater than the right edge of the TCP receiving window. You can **allow** the packets only if the **queue-limit** command is set to 0 (disabled). The default is to drop the packets.
- **synack-data** {**allow** | **drop**}—Allow or drop TCP SYNACK packets that contain data. The default is to drop the packet.
- **syn-data** {**allow** | **drop**}—Allow or drop SYN packets with data. The default is to allow the packet.
- **tcp-options** {**selective-ack** | **timestamp** | **window-scale** | **range** *lower upper*} {**allow** | **clear**}—Set the action for packets with TCP options. Three options are named: **selective-ack** (selective acknowledgment mechanism), **timestamp**, and **window-scale** (window scale mechanism). For other options, you specify them by number on the **range** keyword, where the range limits are 6-7, 9-255. You can enter the command multiple times in a map to define your complete policy.

  You can **allow** the packet (without changing the options), **clear** the options and allow the packet, or **drop** the packet. The default for the three named options is to allow them; the default for all other options is to clear them. Note that clearing the timestamp option disables PAWS and RTT.
- **ttl-evasion-protection**—Protect against TTL evasion attacks. TTL evasion protection is enabled by default, so you would only need to enter the **no** form of this command.

  For example, an attacker can send a packet that passes policy with a very short TTL. When the TTL goes to zero, a router between the ASA and the endpoint drops the packet. It is at this point that the attacker can send a malicious packet with a long TTL that appears to the ASA to be a retransmission and is passed. To the endpoint host, however, it is the first packet that has been received by the attacker. In this case, an attacker is able to succeed without security preventing the attack.
- **urgent-flag** {**allow** | **clear**}—Set the action for packets with the URG flag. You can **allow** the packet, or **clear** the flag and allow the packet. The default is to clear the flag.

The URG flag is used to indicate that the packet contains information that is of higher priority than other data within the stream. The TCP RFC is vague about the exact interpretation of the URG flag, therefore end systems handle urgent offsets in different ways, which may make the end system vulnerable to attacks.

- **window-variation** {**allow** | **drop**}—Allow or drop a connection that has changed its window size unexpectedly. The default is to allow the connection.

  The window size mechanism allows TCP to advertise a large window and to subsequently advertise a much smaller window without having accepted too much data. From the TCP specification, "shrinking the window" is strongly discouraged. When this condition is detected, the connection can be dropped.

Step 3 Apply the TCP map to a traffic class using a service policy.

a. Define the traffic class with an L3/L4 class map and add the map to a policy map.

```
class-map name
match parameter
policy-map name
class name
```

Example:

```
hostname(config)# class-map normalization
hostname(config-cmap)# match any
hostname(config)# policy-map global_policy
hostname(config-pmap)# class normalization
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name. For information on matching statements for class maps, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

b. Apply the TCP map.

set connection advanced-options *tcp-map-name*

Example:

```
hostname(config-pmap-c)# set connection advanced-options tcp_map1
```

c. If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

```
service-policy policymap_name {global | interface interface_name}
```

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Examples

For example, to allow urgent flag and urgent offset packets for all traffic sent to the range of TCP ports between the well known FTP data port and the Telnet port, enter the following commands:

```
hostname(config)# tcp-map tmap
hostname(config-tcp-map)# urgent-flag allow
hostname(config-tcp-map)# class-map urg-class
hostname(config-cmap)# match port tcp range ftp-data telnet
```

```
hostname(config-cmap)# policy-map pmap
hostname(config-pmap)# class urg-class
hostname(config-pmap-c)# set connection advanced-options tmap
hostname(config-pmap-c)# service-policy pmap global
```

Bypass TCP State Checks for Asynchronous Routing (TCP State Bypass)

If you have an asynchronous routing environment in your network, where the outbound and inbound flow for a given connection can go through two different ASA devices, you need to implement TCP State Bypass on the affected traffic.

However, TCP State Bypass weakens the security of your network, so you should apply bypass on very specific, limited traffic classes.

The following topics explain the problem and solution in more detail.

- The Asynchronous Routing Problem, page 16-10
- Guidelines for TCP State Bypass, page 16-11
- Configure TCP State Bypass, page 16-12

The Asynchronous Routing Problem

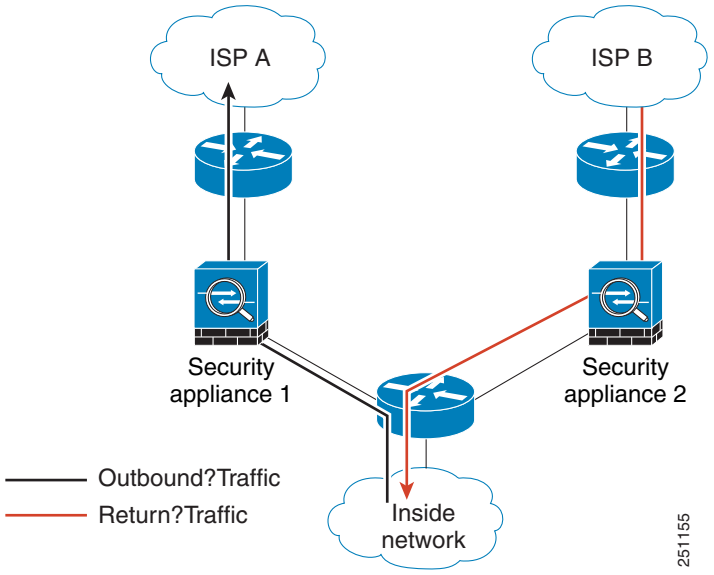
By default, all traffic that goes through the ASA is inspected using the Adaptive Security Algorithm and is either allowed through or dropped based on the security policy. The ASA maximizes the firewall performance by checking the state of each packet (is this a new connection or an established connection?) and assigning it to either the session management path (a new connection SYN packet), the fast path (an established connection), or the control plane path (advanced inspection). See the general operations configuration guide for more detailed information about the stateful firewall.

TCP packets that match existing connections in the fast path can pass through the ASA without rechecking every aspect of the security policy. This feature maximizes performance. However, the method of establishing the session in the fast path using the SYN packet, and the checks that occur in the fast path (such as TCP sequence number), can stand in the way of asymmetrical routing solutions: both the outbound and inbound flow of a connection must pass through the same ASA.

For example, a new connection goes to ASA 1. The SYN packet goes through the session management path, and an entry for the connection is added to the fast path table. If subsequent packets of this connection go through ASA 1, then the packets will match the entry in the fast path, and are passed through. But if subsequent packets go to ASA 2, where there was not a SYN packet that went through

the session management path, then there is no entry in the fast path for the connection, and the packets are dropped. The following figure shows an asymmetric routing example where the outbound traffic goes through a different ASA than the inbound traffic:

Figure 16-1 Asymmetric Routing



If you have asymmetric routing configured on upstream routers, and traffic alternates between two ASAs, then you can configure TCP state bypass for specific traffic. TCP state bypass alters the way sessions are established in the fast path and disables the fast path checks. This feature treats TCP traffic much as it treats a UDP connection: when a non-SYN packet matching the specified networks enters the ASA, and there is not an fast path entry, then the packet goes through the session management path to establish the connection in the fast path. Once in the fast path, the traffic bypasses the fast path checks.

Guidelines for TCP State Bypass

TCP State Bypass Unsupported Features

The following features are not supported when you use TCP state bypass:

- Application inspection—Application inspection requires both inbound and outbound traffic to go through the same ASA, so application inspection is applied TCP state bypass traffic.
- AAA authenticated sessions—When a user authenticates with one ASA, traffic returning via the other ASA will be denied because the user did not authenticate with that ASA.
- TCP Intercept, maximum embryonic connection limit, TCP sequence number randomization—The ASA does not keep track of the state of the connection, so these features are not applied.
- TCP normalization—The TCP normalizer is disabled.
- Service module functionality—You cannot use TCP state bypass and any application running on any type of service module, such as ASA FirePOWER.
- Stateful failover.

TCP State Bypass NAT Guidelines

Because the translation session is established separately for each ASA, be sure to configure static NAT on both ASAs for TCP state bypass traffic. If you use dynamic NAT, the address chosen for the session on ASA 1 will differ from the address chosen for the session on ASA 2.

Configure TCP State Bypass

To bypass TCP state checking in asynchronous routing environments, carefully define a traffic class that applies to the affected hosts or networks only, then enable TCP State Bypass on the traffic class using a service policy. Because bypass reduces the security of the network, limit its application as much as possible.

Procedure

Step 1 Create an L3/L4 class map to identify the hosts that require TCP State Bypass. Use an access-list match to identify the source and destination hosts.

class-map *name*
match *parameter*

Example:

```
hostname(config)# access-list bypass extended permit tcp host 10.1.1.1 host 10.2.2.2
hostname(config)# class-map bypass-class
hostname(config-cmap)# match access-list bypass
```

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic, and identify the class map.

policy-map *name*
class *name*

Example:

```
hostname(config)# policy-map global_policy
hostname(config-pmap)# class bypass-class
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name. For the class map, specify the class you created earlier in this procedure.

Step 3 Enable TCP State Bypass on the class.

set connection advanced-options tcp-state-bypass

Step 4 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Examples

The following is a sample configuration for TCP state bypass:

```
hostname(config)# access-list tcp_bypass extended permit tcp 10.1.1.0 255.255.255.224 any

hostname(config)# class-map tcp_bypass
hostname(config-cmap)# description "TCP traffic that bypasses stateful firewall"
hostname(config-cmap)# match access-list tcp_bypass

hostname(config-cmap)# policy-map tcp_bypass_policy
hostname(config-pmap)# class tcp_bypass
hostname(config-pmap-c)# set connection advanced-options tcp-state-bypass

hostname(config-pmap-c)# service-policy tcp_bypass_policy outside
```

Disable TCP Sequence Randomization

Each TCP connection has two ISNs: one generated by the client and one generated by the server. The ASA randomizes the ISN of the TCP SYN passing in both the inbound and outbound directions.

Randomizing the ISN of the protected host prevents an attacker from predicting the next ISN for a new connection and potentially hijacking the new session.

You can disable TCP initial sequence number randomization if necessary, for example, because data is getting scrambled. For example:

- If another in-line firewall is also randomizing the initial sequence numbers, there is no need for both firewalls to be performing this action, even though this action does not affect the traffic.
- If you use eBGP multi-hop through the ASA, and the eBGP peers are using MD5. Randomization breaks the MD5 checksum.
- You use a WAAS device that requires the ASA not to randomize the sequence numbers of connections.

Procedure

Step 1 Create an L3/L4 class map to identify the traffic whose TCP sequence numbers should not be randomized. The class match should be for TCP traffic; you can identify specific hosts (with an ACL) do a TCP port match, or simply match any traffic.

class-map *name*
match *parameter*

Example:

```
hostname(config)# access-list preserve-sq-no extended permit tcp any host 10.2.2.2
hostname(config)# class-map no-tcp-random
hostname(config-cmap)# match access-list preserve-sq-no
```

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic, and identify the class map.

policy-map *name*
class *name*

Example:

```
hostname(config)# policy-map global_policy
hostname(config-pmap)# class preserve-sq-no
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name. For the class map, specify the class you created earlier in this procedure.

Step 3 Disable TCP sequence number randomization on the class.

```
set connection random-sequence-number disable
```

If you later decide to turn it back on, replace “disable “with **enable**.

Step 4 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Configure Connection Settings for Specific Traffic Classes (All Services)

You can configure different connection settings for specific traffic classes using service policies. Use service policies to:

- Customize connection limits and timeouts used to protect against DoS and SYN-flooding attacks.
- Implement Dead Connection Detection so that valid but idle connections remain alive.
- Disable TCP sequence number randomization in cases where you do not need it.
- Customize how the TCP Normalizer protects against abnormal TCP packets.
- Implement TCP State Bypass for traffic subject to asynchronous routing. Bypass traffic is not subject to inspection.
- Decrement time-to-live (TTL) on packets so that the ASA will show up on trace route output.

Note If you decrement time to live, packets with a TTL of 1 will be dropped, but a connection will be opened for the session on the assumption that the connection might contain packets with a greater TTL. Note that some packets, such as OSPF hello packets, are sent with TTL = 1, so decrementing time to live can have unexpected consequences.

You can configure any combination of these settings for a given traffic class, except for TCP State Bypass and TCP Normalizer customization, which are mutually exclusive.

Tip This procedure shows a service policy for traffic that goes through the ASA. You can also configure the connection maximum and embryonic connection maximum for management (to the box) traffic.

Before You Begin

If you want to customize the TCP Normalizer, create the required TCP Map before proceeding.

The **set connection** command (for connection limits and sequence randomization) and **set connection timeout** commands are described here separately for each parameter. However, you can enter the commands on one line, and if you enter them separately, they are shown in the configuration as one command.

Procedure

Step 1 Create an L3/L4 class map to identify the traffic for which you want to customize connection settings.

```
class-map name
match parameter
```

Example:

```
hostname(config)# class-map CONNS
hostname(config-cmap)# match any
```

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 11-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic, and identify the class map.

```
policy-map name
class name
```

Example:

```
hostname(config)# policy-map global_policy
hostname(config-pmap)# class CONNS
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name. For the class map, specify the class you created earlier in this procedure.

Step 3 Set connection limits and TCP sequence number randomization. (TCP Intercept.)

- **set connection conn-max** n—The maximum number of simultaneous TCP or UDP connections that are allowed, between 0 and 2000000, for the entire class. The default is 0, which allows unlimited connections.
  - If two servers are configured to allow simultaneous TCP or UDP connections, the connection limit is applied to each configured server separately.
  - Because the limit is applied to a class, one attack host can consume all the connections and leave none for the rest of the hosts that are matched to the class.
- **set connection embryonic-conn-max** n—The maximum number of simultaneous embryonic connections allowed, between 0 and 2000000. The default is 0, which allows unlimited connections. By setting a non-zero limit, you enable TCP Intercept, which protects inside systems from a DoS attack perpetrated by flooding an interface with TCP SYN packets. Also set the per-client options to protect against SYN flooding.
- **set connection per-client-embryonic-max** n—The maximum number of simultaneous embryonic connections allowed per client, between 0 and 2000000. The default is 0, which allows unlimited connections.
- **set connection per-client-max** n—The maximum number of simultaneous connections allowed per client, between 0 and 2000000. The default is 0, which allows unlimited connections. This argument restricts the maximum number of simultaneous connections that are allowed for each host that is matched to the class.

- **set connection random-sequence-number** {**enable** | **disable**}—Whether to enable or disable TCP sequence number randomization. Randomization is enabled by default.

Example:

```
hostname(config-pmap-c)# set connection conn-max 256 random-sequence-number disable
```

Step 4 Set connection timeouts and Dead Connection Detection (DCD).

The defaults described below assume you have not changed the global defaults for these behaviors using the **timeout** command; the global defaults override the ones described here. Enter **0** to disable the timer, so that a connection never times out.

- **set connection timeout embryonic** *hh*:*mm*:*ss*—The timeout period until a TCP embryonic (half-open) connection is closed, between 0:0:5 and 1193:00:00. The default is 0:0:30.
- **set connection idle** *hh*:*mm*:*ss* [**reset**]—The idle timeout period after which an established connection of any protocol closes, between 0:0:1 and 1193:0:0. The default is 1:0:0. For TCP traffic, the **reset** keyword sends a reset to TCP endpoints when the connection times out.

  The default **udp** idle timeout is 2 minutes. The default **icmp** idle timeout is 2 seconds. The default **esp** and **ha** idle timeout is 30 seconds. For all other protocols, the default idle timeout is 2 minutes.
- **set connection half-closed** *hh*:*mm*:*ss*—The idle timeout period until a half-closed connection is closed, between 0:5:0 (for 9.1(1) and earlier) or 0:0:30 (for 9.1(2) and later) and 1193:0:0. The default is 0:10:0. Half-closed connections are not affected by DCD. Also, the ASA does not send a reset when taking down half-closed connections.
- **set connection dcd** [*retry-interval* [*max_retries*]]—Enable Dead Connection Detection (DCD). Before expiring an idle connection, the ASA probes the end hosts to determine if the connection is valid. If both hosts respond, the connection is preserved, otherwise the connection is freed.

  The *retry-interval* sets the time duration in *hh***:***mm***:***ss* format to wait after each unresponsive DCD probe before sending another probe, between 0:0:1 and 24:0:0. The default is 0:0:15. The *max-retries* sets the number of consecutive failed retries for DCD before declaring the connection as dead. The minimum value is 1 and the maximum value is 255. The default is 5.

Example:

```
hostname(config-pmap-c)# set connection timeout idle 2:0:0 embryonic 0:40:0
half-closed 0:20:0 dcd
```

Step 5 Decrement time-to-live (TTL) on packets that match the class.

set connection decrement-ttl

This command, along with the **icmp unreachable** command, is required to allow a traceroute through the ASA that shows the ASA as one of the hops.

Example:

```
hostname(config)# class-map global-policy
hostname(config-cmap)# match any
hostname(config-cmap)# exit
hostname(config)# policy-map global_policy
hostname(config-pmap)# class global-policy
hostname(config-pmap-c)# set connection decrement-ttl
hostname(config-pmap-c)# exit
hostname(config)# icmp unreachable rate-limit 50 burst-size 6
```

Step 6 Customize TCP Normalizer behavior by applying a TCP map.

set connection advanced-options *tcp-map-name*

Example:

```
hostname(config-pmap-c)# set connection advanced-options tcp_map1
```

Step 7 Implement TCP State Bypass.

set connection advanced-options tcp-state-bypass

Step 8 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Examples

The following example sets the connection limits and timeouts for all traffic:

```
hostname(config)# class-map CONNS
hostname(config-cmap)# match any
hostname(config-cmap)# policy-map CONNS
hostname(config-pmap)# class CONNS
hostname(config-pmap-c)# set connection conn-max 1000 embryonic-conn-max 3000
hostname(config-pmap-c)# set connection timeout idle 2:0:0 embryonic 0:40:0
half-closed 0:20:0 dcd
hostname(config-pmap-c)# service-policy CONNS interface outside
```

You can enter **set connection** commands with multiple parameters or you can enter each parameter as a separate command. The ASA combines the commands into one line in the running configuration. For example, if you entered the following two commands in class configuration mode:

```
hostname(config-pmap-c)# set connection conn-max 600
hostname(config-pmap-c)# set connection embryonic-conn-max 50
```

The output of the **show running-config policy-map** command would display the result of the two commands in a single, combined command:

```
set connection conn-max 600 embryonic-conn-max 50
```

Monitoring Connections

You can use the following commands to monitor connections:

- **show conn**

  Shows connection information. The “b” flag indicates traffic subject to TCP State Bypass.

- **show service-policy**

  Shows service policy statistics, including Dead Connection Detection (DCD) statistics.

- **show threat-detection statistics top tcp-intercept** [**all** | **detail**]

  View the top 10 protected servers under attack. The **all** keyword shows the history data of all the traced servers. The **detail** keyword shows history sampling data. The ASA samples the number of attacks 30 times during the rate interval, so for the default 30 minute period, statistics are collected every 60 seconds.

History for Connection Settings

Feature Name	Platform Releases	Description
TCP state bypass	8.2(1)	This feature was introduced. The following command was introduced: **set connection advanced-options tcp-state-bypass**.
Connection timeout for all protocols	8.2(2)	The idle timeout was changed to apply to all protocols, not just TCP. The following command was modified: **set connection timeout**
Timeout for connections using a backup static route	8.2(5)/8.4(2)	When multiple static routes exist to a network with different metrics, the ASA uses the one with the best metric at the time of connection creation. If a better route becomes available, then this timeout lets connections be closed so a connection can be reestablished to use the better route. The default is 0 (the connection never times out). To take advantage of this feature, change the timeout to a new value. We modified the following command: **timeout floating-conn**.
Configurable timeout for PAT xlate	8.4(3)	When a PAT xlate times out (by default after 30 seconds), and the ASA reuses the port for a new translation, some upstream routers might reject the new connection because the previous connection might still be open on the upstream device. The PAT xlate timeout is now configurable, to a value between 30 seconds and 5 minutes. We introduced the following command: **timeout pat-xlate**. *This feature is not available in 8.5(1) or 8.6(1).*

Feature Name	Platform Releases	Description
Increased maximum connection limits for service policy rules	9.0(1)	The maximum number of connections for service policy rules was increased from 65535 to 2000000. We modified the following commands: **set connection conn-max**, **set connection embryonic-conn-max**, **set connection per-client-embryonic-max**, **set connection per-client-max**.
Decreased the half-closed timeout minimum value to 30 seconds	9.1(2)	The half-closed timeout minimum value for both the global timeout and connection timeout was lowered from 5 minutes to 30 seconds to provide better DoS protection. We modified the following commands: **set connection timeout half-closed**, **timeout half-closed**.

Quality of Service

Have you ever participated in a long-distance phone call that involved a satellite connection? The conversation might be interrupted with brief, but perceptible, gaps at odd intervals. Those gaps are the time, called the latency, between the arrival of packets being transmitted over the network. Some network traffic, such as voice and video, cannot tolerate long latency times. Quality of service (QoS) is a feature that lets you give priority to critical traffic, prevent bandwidth hogging, and manage network bottlenecks to prevent packet drops.

Note For the ASASM, we suggest performing QoS on the switch instead of the ASASM. Switches have more capability in this area. In general, QoS is best performed on the routers and switches in the network, which tend to have more extensive capabilities than the ASA.

This chapter describes how to apply QoS policies.

About QoS

You should consider that in an ever-changing network environment, QoS is not a one-time deployment, but an ongoing, essential part of network design.

This section describes the QoS features available on the ASA.

Supported QoS Features

The ASA supports the following QoS features:

- Policing—To prevent classified traffic from hogging the network bandwidth, you can limit the maximum bandwidth used per class. See Policing, page 17-2 for more information.
- Priority queuing—For critical traffic that cannot tolerate latency, such as Voice over IP (VoIP), you can identify traffic for Low Latency Queuing (LLQ) so that it is always transmitted ahead of other traffic. See Priority Queuing, page 17-3.

What is a Token Bucket?

A token bucket is used to manage a device that regulates the data in a flow, for example, a traffic policer. A token bucket itself has no discard or priority policy. Rather, a token bucket discards tokens and leaves to the flow the problem of managing its transmission queue if the flow overdrives the regulator.

A token bucket is a formal definition of a rate of transfer. It has three components: a burst size, an average rate, and a time interval. Although the average rate is generally represented as bits per second, any two values may be derived from the third by the relation shown as follows:

average rate = burst size / time interval

Here are some definitions of these terms:

- Average rate—Also called the committed information rate (CIR), it specifies how much data can be sent or forwarded per unit time on average.
- Burst size—Also called the Committed Burst (Bc) size, it specifies in bytes per burst how much traffic can be sent within a given unit of time to not create scheduling concerns.
- Time interval—Also called the measurement interval, it specifies the time quantum in seconds per burst.

In the token bucket metaphor, tokens are put into the bucket at a certain rate. The bucket itself has a specified capacity. If the bucket fills to capacity, newly arriving tokens are discarded. Each token is permission for the source to send a certain number of bits into the network. To send a packet, the regulator must remove from the bucket a number of tokens equal in representation to the packet size.

If not enough tokens are in the bucket to send a packet, the packet waits until the packet is discarded or marked down. If the bucket is already full of tokens, incoming tokens overflow and are not available to future packets. Thus, at any time, the largest burst a source can send into the network is roughly proportional to the size of the bucket.

Policing

Policing is a way of ensuring that no traffic exceeds the maximum rate (in bits/second) that you configure, thus ensuring that no one traffic class can take over the entire resource. When traffic exceeds the maximum rate, the ASA drops the excess traffic. Policing also sets the largest single burst of traffic allowed.

Priority Queuing

LLQ priority queuing lets you prioritize certain traffic flows (such as latency-sensitive traffic like voice and video) ahead of other traffic. Priority queuing uses an LLQ priority queue on an interface (see Configure the Priority Queue for an Interface, page 17-6), while all other traffic goes into the "best effort" queue. Because queues are not of infinite size, they can fill and overflow. When a queue is full, any additional packets cannot get into the queue and are dropped. This is called *tail drop*. To avoid having the queue fill up, you can increase the queue buffer size. You can also fine-tune the maximum number of packets allowed into the transmit queue. These options let you control the latency and robustness of the priority queuing. Packets in the LLQ queue are always transmitted before packets in the best effort queue.

How QoS Features Interact

You can configure each of the QoS features alone if desired for the ASA. Often, though, you configure multiple QoS features on the ASA so you can prioritize some traffic, for example, and prevent other traffic from causing bandwidth problems. You can configure:

Priority queuing (for specific traffic) + Policing (for the rest of the traffic).

You cannot configure priority queuing and policing for the same set of traffic.

DSCP (DiffServ) Preservation

DSCP (DiffServ) markings are preserved on all traffic passing through the ASA. The ASA does not locally mark/remark any classified traffic. For example, you could key off the Expedited Forwarding (EF) DSCP bits of every packet to determine if it requires "priority" handling and have the ASA direct those packets to the LLQ.

Guidelines for QoS

Context Mode Guidelines

Supported in single context mode only. Does not support multiple context mode.

Firewall Mode Guidelines

Supported in routed firewall mode only. Does not support transparent firewall mode.

IPv6 Guidelines

Does not support IPv6.

Model Guidelines

- (ASA 5512-X through ASA 5555-X) Priority queuing is not supported on the Management 0/0 interface.
- (ASASM) Only policing is supported.

Additional Guidelines and Limitations

- QoS is applied unidirectionally; only traffic that enters (or exits, depending on the QoS feature) the interface to which you apply the policy map is affected. See Feature Directionality, page 11-4 for more information.
- For priority traffic, you cannot use the **class-default** class map.
- For priority queuing, the priority queue must be configured for a physical interface or, for the ASASM, a VLAN.
- For policing, to-the-box traffic is not supported.
- For policing, traffic to and from a VPN tunnel bypasses interface policing.
- For policing, when you match a tunnel group class map, only outbound policing is supported.

Configure QoS

Use the following sequence to implement QoS on the ASA.

Step 1 Determine the Queue and TX Ring Limits for a Priority Queue, page 17-4.

Step 2 Configure the Priority Queue for an Interface, page 17-6.

Step 3 Configure a Service Rule for Priority Queuing and Policing, page 17-7.

Determine the Queue and TX Ring Limits for a Priority Queue

Use the following worksheets to determine the priority queue and TX ring limits.

- Queue Limit Worksheet, page 17-4
- TX Ring Limit Worksheet, page 17-5

Queue Limit Worksheet

The following worksheet shows how to calculate the priority queue size. Because queues are not of infinite size, they can fill and overflow. When a queue is full, any additional packets cannot get into the queue and are dropped (called *tail drop*). To avoid having the queue fill up, you can adjust the queue buffer size according to Configure the Priority Queue for an Interface, page 17-6.

Tips on the worksheet:

- Outbound bandwidth—For example, DSL might have an uplink speed of 768 Kbps. Check with your provider.
- Average packet size—Determine this value from a codec or sampling size. For example, for VoIP over VPN, you might use 160 bytes. We recommend 256 bytes if you do not know what size to use.
- Delay—The delay depends on your application. For example, the recommended maximum delay for VoIP is 200 ms. We recommend 500 ms if you do not know what delay to use.

Table 17-1 Queue Limit Worksheet

1	_____ *Outbound bandwidth (Mbps or Kbps)*	Mbps	x	**125**	=	_____ *# of bytes/ms*		
		Kbps	x	**.125**	=	_____ *# of bytes/ms*		
2	_____ *# of bytes/ms from Step 1*		÷	_____ *Average packet size (bytes)*	x	_____ *Delay (ms)*	=	_____ *Queue limit (# of packets)*

TX Ring Limit Worksheet

The following worksheet shows how to calculate the TX ring limit. This limit determines the maximum number of packets allowed into the Ethernet transmit driver before the driver pushes back to the queues on the interface to let them buffer packets until the congestion clears. This setting guarantees that the hardware-based transmit ring imposes a limited amount of extra latency for a high-priority packet.

Tips on the worksheet:

- Outbound bandwidth—For example, DSL might have an uplink speed of 768 Kbps. Check with your provider.
- Maximum packet size—Typically, the maximum size is 1538 bytes, or 1542 bytes for tagged Ethernet. If you allow jumbo frames (if supported for your platform), then the packet size might be larger.
- Delay—The delay depends on your application. For example, to control jitter for VoIP, you should use 20 ms.

Table 17-2 TX Ring Limit Worksheet

1	_____ *Outbound bandwidth (Mbps or Kbps)*	Mbps	x	**125**	=	_____ *# of bytes/ms*		
		Kbps	x	**0.125**	=	_____ *# of bytes/ms*		
2	_____ *# of bytes/ms from Step 1*		÷	_____ *Maximum packet size (bytes)*	x	_____ *Delay (ms)*	=	_____ *TX ring limit (# of packets)*

Configure the Priority Queue for an Interface

If you enable priority queuing for traffic on a physical interface, then you need to also create the priority queue on each interface. Each physical interface uses two queues: one for priority traffic, and the other for all other traffic. For the other traffic, you can optionally configure policing.

Before You Begin

- (ASASM) The ASASM does not support priority queuing.
- (ASA 5512-X through ASA 5555-X) Priority queuing is not supported on the Management 0/0 interface.

Procedure

Step 1 Create the priority queue for the interface.

priority-queue *interface_name*

Example:

```
hostname(config)# priority-queue inside
```

The *interface_name* argument specifies the physical interface name on which you want to enable the priority queue, or for the ASASM, the VLAN interface name.

Step 2 Change the size of the priority queues.

queue-limit *number_of_packets*

Example:

```
hostname(config-priority-queue)# queue-limit 260
```

The default queue limit is 1024 packets. Because queues are not of infinite size, they can fill and overflow. When a queue is full, any additional packets cannot get into the queue and are dropped (called *tail drop*). To avoid having the queue fill up, you can use the **queue-limit** command to increase the queue buffer size.

The upper limit of the range of values for the **queue-limit** command is determined dynamically at run time. To view this limit, enter **queue-limit ?** on the command line. The key determinants are the memory needed to support the queues and the memory available on the device.

The **queue-limit** that you specify affects both the higher priority low-latency queue and the best effort queue.

Step 3 Specify the depth of the priority queues.

tx-ring-limit *number_of_packets*

Example:

```
hostname(config-priority-queue)# tx-ring-limit 3
```

The default tx-ring-limit is 128 packets. This command sets the maximum number of low-latency or normal priority packets allowed into the Ethernet transmit driver before the driver pushes back to the queues on the interface to let them buffer packets until the congestion clears. This setting guarantees that the hardware-based transmit ring imposes a limited amount of extra latency for a high-priority packet.

The upper limit of the range of values for the **tx-ring-limit** command is determined dynamically at run time. To view this limit, enter **tx-ring-limit ?** on the command line. The key determinants are the memory needed to support the queues and the memory available on the device.

The **tx-ring-limit** that you specify affects both the higher priority low-latency queue and the best-effort queue.

Examples

The following example establishes a priority queue on interface “outside” (the GigabitEthernet0/1 interface), with the default queue-limit and tx-ring-limit:

```
hostname(config)# priority-queue outside
```

The following example establishes a priority queue on the interface “outside” (the GigabitEthernet0/1 interface), sets the queue-limit to 260 packets, and sets the tx-ring-limit to 3:

```
hostname(config)# priority-queue outside
hostname(config-priority-queue)# queue-limit 260
hostname(config-priority-queue)# tx-ring-limit 3
```

Configure a Service Rule for Priority Queuing and Policing

You can configure priority queuing and policing for different class maps within the same policy map. See How QoS Features Interact, page 17-3 for information about valid QoS configurations.

Before You Begin

- You cannot use the **class-default** class map for priority traffic.
- (ASASM) The ASASM only supports policing.
- For policing, to-the-box traffic is not supported.
- For policing, traffic to and from a VPN tunnel bypasses interface policing.
- For policing, when you match a tunnel group class map, only outbound policing is supported.
- For priority traffic, identify only latency-sensitive traffic.
- For policing traffic, you can choose to police all other traffic, or you can limit the traffic to certain types.

Procedure

Step 1 Create a class map to identify the traffic for which you want to perform priority queuing.

class-map *priority_map_name*

Example:

```
hostname(config)# class-map priority_traffic
```

Step 2 Specify the traffic in the class map.

match *parameter*

Example:

```
hostname(config-cmap)# match access-list priority
```

See Identify Traffic (Layer 3/4 Class Maps), page 11-13 for more information.

Step 3 Create a class map to identify the traffic for which you want to perform policing.

class-map *policing_map_name*

Example:

```
hostname(config)# class-map policing_traffic
```

Step 4 Specify the traffic in the class map.

match *parameter*

Example:

```
hostname(config-cmap)# match access-list policing
```

See Identify Traffic (Layer 3/4 Class Maps), page 11-13 for more information.

Tip If you use an ACL for traffic matching, policing is applied in the direction specified in the ACL only. That is, traffic going from the source to the destination is policed, but not the reverse.

Step 5 Add or edit a policy map.

policy-map *name*

Example:

```
hostname(config)# policy-map QoS_policy
```

Step 6 Identify the class map you created for prioritized traffic.

class *priority_map_name*

Example:

```
hostname(config-pmap)# class priority_class
```

Step 7 Configure priority queuing for the class.

priority

Example:

```
hostname(config-pmap-c)# priority
```

Step 8 Identify the class map you created for policed traffic.

class *policing_map_name*

Example:

```
hostname(config-pmap)# class policing_class
```

Step 9 Configure policing for the class.

police {**output** | **input**} *conform-rate* [*conform-burst*] [**conform-action** [**drop** | **transmit**]] [**exceed-action** [**drop** | **transmit**]]

Example:

```
hostname(config-pmap-c)# police output 56000 10500
```

The options are:

- *conform-burst argument*—Specifies the maximum number of instantaneous bytes allowed in a sustained burst before throttling to the conforming rate value, between 1000 and 512000000 bytes.

- **conform-action**—Sets the action to take when the rate is less than the *conform_burst* value.
- *conform-rate*—Sets the rate limit for this traffic class; between 8000 and 2000000000 bits per second.]
- **drop**—Drops the packet.
- **exceed-action**—Sets the action to take when the rate is between the *conform-rate* value and the *conform-burst* value.
- **input**—Enables policing of traffic flowing in the input direction.
- **output**—Enables policing of traffic flowing in the output direction.
- **transmit**—Transmits the packet.

Step 10 Activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy QoS_policy interface inside
```

The **global** option applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Monitor QoS

QoS Police Statistics

To view the QoS statistics for traffic policing, use the **show service-policy police** command.

```
hostname# show service-policy police

Global policy:
    Service-policy: global_fw_policy

Interface outside:
    Service-policy: qos
        Class-map: browse
            police Interface outside:
                cir 56000 bps, bc 10500 bytes
                conformed 10065 packets, 12621510 bytes; actions: transmit
                exceeded 499 packets, 625146 bytes; actions: drop
                conformed 5600 bps, exceed 5016 bps
        Class-map: cmap2
            police Interface outside:
                cir 200000 bps, bc 37500 bytes
                conformed 17179 packets, 20614800 bytes; actions: transmit
                exceeded 617 packets, 770718 bytes; actions: drop
```

```
conformed 198785 bps, exceed 2303 bps
```

QoS Priority Statistics

To view statistics for service policies implementing the **priority** command, use the **show service-policy priority** command.

```
hostname# show service-policy priority
Global policy:
    Service-policy: global_fw_policy
Interface outside:
    Service-policy: qos
        Class-map: TG1-voice
            Priority:
                Interface outside: aggregate drop 0, aggregate transmit 9383
```

“Aggregate drop” denotes the aggregated drop in this interface; “aggregate transmit” denotes the aggregated number of transmitted packets in this interface.

QoS Priority Queue Statistics

To display the priority-queue statistics for an interface, use the **show priority-queue statistics** command. The results show the statistics for both the best-effort (BE) queue and the low-latency queue (LLQ). The following example shows the use of the **show priority-queue statistics** command for the interface named test.

```
hostname# show priority-queue statistics test

Priority-Queue Statistics interface test

Queue Type         = BE
Packets Dropped    = 0
Packets Transmit   = 0
Packets Enqueued   = 0
Current Q Length   = 0
Max Q Length       = 0

Queue Type         = LLQ
Packets Dropped    = 0
Packets Transmit   = 0
Packets Enqueued   = 0
Current Q Length   = 0
Max Q Length       = 0
hostname#
```

In this statistical report:

- “Packets Dropped” denotes the overall number of packets that have been dropped in this queue.
- “Packets Transmit” denotes the overall number of packets that have been transmitted in this queue.
- “Packets Enqueued” denotes the overall number of packets that have been queued in this queue.
- “Current Q Length” denotes the current depth of this queue.
- “Max Q Length” denotes the maximum depth that ever occurred in this queue.

Configuration Examples for Priority Queuing and Policing

The following sections provide examples of configuring priority queuing and policing.

Class Map Examples for VPN Traffic

In the following example, the **class-map** command classifies all non-tunneled TCP traffic, using an ACL named tcp_traffic:

```
hostname(config)# access-list tcp_traffic permit tcp any any
hostname(config)# class-map tcp_traffic
hostname(config-cmap)# match access-list tcp_traffic
```

In the following example, other, more specific match criteria are used for classifying traffic for specific, security-related tunnel groups. These specific match criteria stipulate that a match on tunnel-group (in this case, the previously-defined Tunnel-Group-1) is required as the first match characteristic to classify traffic for a specific tunnel, and it allows for an additional match line to classify the traffic (IP differential services code point, expedited forwarding).

```
hostname(config)# class-map TG1-voice
hostname(config-cmap)# match tunnel-group tunnel-grp1
hostname(config-cmap)# match dscp ef
```

In the following example, the **class-map** command classifies both tunneled and non-tunneled traffic according to the traffic type:

```
hostname(config)# access-list tunneled extended permit ip 10.10.34.0 255.255.255.0
192.168.10.0 255.255.255.0
hostname(config)# access-list non-tunneled extended permit tcp any any
hostname(config)# tunnel-group tunnel-grp1 type IPsec_L2L

hostname(config)# class-map browse
hostname(config-cmap)# description "This class-map matches all non-tunneled tcp traffic."
hostname(config-cmap)# match access-list non-tunneled

hostname(config-cmap)# class-map TG1-voice
hostname(config-cmap)# description "This class-map matches all dscp ef traffic for
tunnel-grp 1."
hostname(config-cmap)# match dscp ef
hostname(config-cmap)# match tunnel-group tunnel-grp1

hostname(config-cmap)# class-map TG1-BestEffort
hostname(config-cmap)# description "This class-map matches all best-effort traffic for
tunnel-grp1."
hostname(config-cmap)# match tunnel-group tunnel-grp1
hostname(config-cmap)# match flow ip destination-address
```

The following example shows a way of policing traffic within a tunnel, provided the classed traffic is not specified as a tunnel, but does go *through* the tunnel. In this example, 192.168.10.10 is the address of the host machine on the private side of the remote tunnel, and the ACL is named “host-over-l2l”. By creating a class-map (named “host-specific”), you can then police the “host-specific” class before the LAN-to-LAN connection polices the tunnel. In this example, the “host-specific” traffic is rate-limited before the tunnel, then the tunnel is rate-limited:

```
hostname(config)# access-list host-over-l2l extended permit ip any host 192.168.10.10
hostname(config)# class-map host-specific
hostname(config-cmap)# match access-list host-over-l2l
```

Priority and Policing Example

The following example builds on the configuration developed in the previous section. As in the previous example, there are two named class-maps: tcp_traffic and TG1-voice.

```
hostname(config)# class-map TG1-best-effort
hostname(config-cmap)# match tunnel-group Tunnel-Group-1
hostname(config-cmap)# match flow ip destination-address
```

Adding a third class map provides a basis for defining a tunneled and non-tunneled QoS policy, as follows, which creates a simple QoS policy for tunneled and non-tunneled traffic, assigning packets of the class TG1-voice to the low latency queue and setting rate limits on the tcp_traffic and TG1-best-effort traffic flows.

In this example, the maximum rate for traffic of the tcp_traffic class is 56,000 bits/second and a maximum burst size of 10,500 bytes per second. For the TC1-BestEffort class, the maximum rate is 200,000 bits/second, with a maximum burst of 37,500 bytes/second. Traffic in the TC1-voice class has no policed maximum speed or burst rate because it belongs to a priority class.

```
hostname(config)# access-list tcp_traffic permit tcp any any
hostname(config)# class-map tcp_traffic
hostname(config-cmap)# match access-list tcp_traffic

hostname(config)# class-map TG1-voice
hostname(config-cmap)# match tunnel-group tunnel-grp1
hostname(config-cmap)# match dscp ef

hostname(config-cmap)# class-map TG1-BestEffort
hostname(config-cmap)# match tunnel-group tunnel-grp1
hostname(config-cmap)# match flow ip destination-address

hostname(config)# policy-map qos
hostname(config-pmap)# class tcp_traffic
hostname(config-pmap-c)# police output 56000 10500

hostname(config-pmap-c)# class TG1-voice
hostname(config-pmap-c)# priority

hostname(config-pmap-c)# class TG1-best-effort
hostname(config-pmap-c)# police output 200000 37500

hostname(config-pmap-c)# class class-default
hostname(config-pmap-c)# police output 1000000 37500

hostname(config-pmap-c)# service-policy qos global
```

History for QoS

Feature Name	Platform Releases	Description
Priority queuing and policing	7.0(1)	We introduced QoS priority queuing and policing. We introduced the following commands: **priority-queue**, **queue-limit**, **tx-ring-limit**, **priority, police, show priority-queue statistics, show service-policy police, show service-policy priority, show running-config priority-queue, clear configure priority-queue** .
Shaping and hierarchical priority queuing	7.2(4)/8.0(4)	We introduced QoS shaping and hierarchical priority queuing. We introduced the following commands: **shape**, **show service-policy shape**.
Ten Gigabit Ethernet support for a standard priority queue on the ASA 5585-X	8.2(3)/8.4(1)	We added support for a standard priority queue on Ten Gigabit Ethernet interfaces for the ASA 5585-X.

Threat Detection

This chapter describes how to configure threat detection statistics and scanning threat detection.

Detecting Threats

Threat detection on the ASA provides a front-line defense against attacks. Threat detection works at Layer 3 and 4 to develop a baseline for traffic on the device, analyzing packet drop statistics and accumulating “top” reports based on traffic patterns. In comparison, a module that provides IPS or Next Generation IPS services identifies and mitigates attack vectors up to Layer 7 on traffic the ASA permitted, and cannot see the traffic dropped already by the ASA. Thus, threat detection and IPS can work together to provide a more comprehensive threat defense.

Threat detection consists of the following elements:

- Different levels of statistics gathering for various threats.

  Threat detection statistics can help you manage threats to your ASA; for example, if you enable scanning threat detection, then viewing statistics can help you analyze the threat. You can configure two types of threat detection statistics:

  - Basic threat detection statistics—Includes information about attack activity for the system as a whole. Basic threat detection statistics are enabled by default and have no performance impact.
  - Advanced threat detection statistics—Tracks activity at an object level, so the ASA can report activity for individual hosts, ports, protocols, or ACLs. Advanced threat detection statistics can have a major performance impact, depending on the statistics gathered, so only the ACL statistics are enabled by default.
- Scanning threat detection, which determines when a host is performing a scan. You can optionally shun any hosts determined to be a scanning threat.

Basic Threat Detection Statistics

Using basic threat detection statistics, the ASA monitors the rate of dropped packets and security events due to the following reasons:

- Denial by ACLs.
- Bad packet format (such as invalid-ip-header or invalid-tcp-hdr-length).
- Connection limits exceeded (both system-wide resource limits, and limits set in the configuration).
- DoS attack detected (such as an invalid SPI, Stateful Firewall check failure).
- Basic firewall checks failed. This option is a combined rate that includes all firewall-related packet drops in this list. It does not include non-firewall-related drops such as interface overload, packets failed at application inspection, and scanning attack detected.
- Suspicious ICMP packets detected.
- Packets failed application inspection.
- Interface overload.
- Scanning attack detected. This option monitors scanning attacks; for example, the first TCP packet is not a SYN packet, or the TCP connection failed the 3-way handshake. Full scanning threat detection takes this scanning attack rate information and acts on it by classifying hosts as attackers and automatically shunning them, for example.
- Incomplete session detection such as TCP SYN attack detected or UDP session with no return data attack detected.

When the ASA detects a threat, it immediately sends a system log message (733100). The ASA tracks two types of rates: the average event rate over an interval, and the burst event rate over a shorter burst interval. The burst rate interval is 1/30th of the average rate interval or 10 seconds, whichever is higher. For each received event, the ASA checks the average and burst rate limits; if both rates are exceeded, then the ASA sends two separate system messages, with a maximum of one message for each rate type per burst period.

Basic threat detection affects performance only when there are drops or potential threats; even in this scenario, the performance impact is insignificant.

Advanced Threat Detection Statistics

Advanced threat detection statistics show both allowed and dropped traffic rates for individual objects such as hosts, ports, protocols, or ACLs.

Caution Enabling advanced statistics can affect the ASA performance, depending on the type of statistics enabled. The **threat-detection statistics host** command affects performance in a significant way; if you have a high traffic load, you might consider enabling this type of statistics temporarily. The **threat-detection statistics port** command, however, has modest impact.

Scanning Threat Detection

A typical scanning attack consists of a host that tests the accessibility of every IP address in a subnet (by scanning through many hosts in the subnet or sweeping through many ports in a host or subnet). The scanning threat detection feature determines when a host is performing a scan. Unlike IPS scan detection that is based on traffic signatures, ASA threat detection scanning maintains an extensive database that contains host statistics that can be analyzed for scanning activity.

The host database tracks suspicious activity such as connections with no return activity, access of closed service ports, vulnerable TCP behaviors such as non-random IPID, and many more behaviors.

If the scanning threat rate is exceeded, then the ASA sends a syslog message (733101), and optionally shuns the attacker. The ASA tracks two types of rates: the average event rate over an interval, and the burst event rate over a shorter burst interval. The burst event rate is 1/30th of the average rate interval or 10 seconds, whichever is higher. For each event detected that is considered to be part of a scanning attack, the ASA checks the average and burst rate limits. If either rate is exceeded for traffic sent from a host, then that host is considered to be an attacker. If either rate is exceeded for traffic received by a host, then that host is considered to be a target.

The following table lists the default rate limits for scanning threat detection.

Table 18-1 Default Rate Limits for Scanning Threat Detection

Average Rate	Burst Rate
5 drops/sec over the last 600 seconds.	10 drops/sec over the last 20 second period.
5 drops/sec over the last 3600 seconds.	10 drops/sec over the last 120 second period.

Caution The scanning threat detection feature can affect the ASA performance and memory significantly while it creates and gathers host- and subnet-based data structure and information.

Guidelines for Threat Detection

Security Context Guidelines

Except for advanced threat statistics, threat detection is supported in single mode only. In Multiple mode, TCP Intercept statistics are the only statistic supported.

Firewall Mode Guidelines

Supported in routed and transparent firewall mode.

Types of Traffic Monitored

- Only through-the-box traffic is monitored; to-the-box traffic is not included in threat detection.
- Traffic that is denied by an ACL does not trigger scanning threat detection; only traffic that is allowed through the ASA and that creates a flow is affected by scanning threat detection.

Defaults for Threat Detection

Basic threat detection statistics are enabled by default.

The following table lists the default settings. You can view all these default settings using the **show running-config all threat-detection** command.

For advanced statistics, by default, statistics for ACLs are enabled.

Table 18-2 Basic Threat Detection Default Settings

	Trigger Settings	
Packet Drop Reason	**Average Rate**	**Burst Rate**
• DoS attack detected • Bad packet format • Connection limits exceeded • Suspicious ICMP packets detected	100 drops/sec over the last 600 seconds.	400 drops/sec over the last 20 second period.
	80 drops/sec over the last 3600 seconds.	320 drops/sec over the last 120 second period.
Scanning attack detected	5 drops/sec over the last 600 seconds.	10 drops/sec over the last 20 second period.
	4 drops/sec over the last 3600 seconds.	8 drops/sec over the last 120 second period.
Incomplete session detected such as TCP SYN attack detected or UDP session with no return data attack detected (combined)	100 drops/sec over the last 600 seconds.	200 drops/sec over the last 20 second period.
	80 drops/sec over the last 3600 seconds.	160 drops/sec over the last 120 second period.
Denial by ACLs	400 drops/sec over the last 600 seconds.	800 drops/sec over the last 20 second period.
	320 drops/sec over the last 3600 seconds.	640 drops/sec over the last 120 second period.
• Basic firewall checks failed • Packets failed application inspection	400 drops/sec over the last 600 seconds.	1600 drops/sec over the last 20 second period.
	320 drops/sec over the last 3600 seconds.	1280 drops/sec over the last 120 second period.
Interface overload	2000 drops/sec over the last 600 seconds.	8000 drops/sec over the last 20 second period.
	1600 drops/sec over the last 3600 seconds.	6400 drops/sec over the last 120 second period.

Configure Threat Detection

Basic threat detection statistics are enabled by default, and might be the only threat detection service that you need. Use the following procedure if you want to implement additional threat detection services.

Procedure

Step 1 Configure Basic Threat Detection Statistics, page 18-5.

Basic threat detection statistics include activity that might be related to an attack, such as a DoS attack.

Step 2 Configure Advanced Threat Detection Statistics, page 18-5.

Step 3 Configure Scanning Threat Detection, page 18-7.

Configure Basic Threat Detection Statistics

Basic threat detection statistics is enabled by default. You can disabled it, or turn it on again if you disable it.

Procedure

Step 1 Enable basic threat detection statistics (if you previously disabled it).

```
threat-detection basic-threat
```

Example:

```
hostname(config)# threat-detection basic-threat
```

Basic threat detection is enabled by default. Use **no threat-detection basic-threat** to disable it.

Step 2 (Optional) Change the default settings for one or more type of event.

```
threat-detection rate {acl-drop | bad-packet-drop | conn-limit-drop | dos-drop |
fw-drop | icmp-drop | inspect-drop | interface-drop | scanning-threat | syn-attack}
rate-interval rate_interval average-rate av_rate burst-rate burst_rate
```

Example:

```
hostname(config)# threat-detection rate dos-drop rate-interval 600 average-rate 60
burst-rate 100
```

For a description of each event type, see Basic Threat Detection Statistics, page 18-2.

When you use this command with the **scanning-threat** keyword, it is also used in the scanning threat detection. If you do not configure basic threat detection, you can still use this command with the **scanning-threat** keyword to configure the rate limits for scanning threat detection.

You can configure up to three different rate intervals for each event type.

Configure Advanced Threat Detection Statistics

You can configure the ASA to collect extensive statistics. By default, statistics for ACLs are enabled. To enable other statistics, perform the following steps.

Procedure

Step 1 (Optional) Enable *all* statistics.

```
threat-detection statistics
```

Example:

```
hostname(config)# threat-detection statistics
```

To enable only certain statistics, enter this command for each statistic type (shown in this table), and do not also enter the command without any options. You can enter **threat-detection statistics** (without any options) and then customize certain statistics by entering the command with statistics-specific options (for example, **threat-detection statistics host number-of-rate 2**). If you enter **threat-detection statistics** (without any options) and then enter a command for specific statistics, but without any statistic-specific options, then that command has no effect because it is already enabled.

If you enter the **no** form of this command, it removes all **threat-detection statistics** commands, including the **threat-detection statistics access-list** command, which is enabled by default.

Step 2 (Optional) Enable statistics for ACLs (if they were disabled previously).

```
threat-detection statistics access-list
```

Example:

```
hostname(config)# threat-detection statistics access-list
```

Statistics for ACLs are enabled by default. ACL statistics are only displayed using the **show threat-detection top access-list** command. This command is enabled by default.

Step 3 (Optional) Configure statistics for hosts (**host** keyword), TCP and UDP ports (**port** keyword), or non-TCP/UDP IP protocols (**protocol** keyword).

```
threat-detection statistics {host | port | protocol} [number-of-rate {1 | 2 | 3}]
```

Example:

```
hostname(config)# threat-detection statistics host number-of-rate 2

hostname(config)# threat-detection statistics port number-of-rate 2

hostname(config)# threat-detection statistics protocol number-of-rate 3
```

The **number-of-rate** keyword sets the number of rate intervals maintained for statistics. The default number of rate intervals is **1**, which keeps the memory usage low. To view more rate intervals, set the value to **2** or **3**. For example, if you set the value to **3**, then you view data for the last 1 hour, 8 hours, and 24 hours. If you set this keyword to **1** (the default), then only the shortest rate interval statistics are maintained. If you set the value to **2**, then the two shortest intervals are maintained.

The host statistics accumulate for as long as the host is active and in the scanning threat host database. The host is deleted from the database (and the statistics cleared) after 10 minutes of inactivity.

Step 4 (Optional) Configure statistics for attacks intercepted by TCP Intercept (to enable TCP Intercept, see Protect Servers from a SYN Flood DoS Attack (TCP Intercept), page 16-4).

```
threat-detection statistics tcp-intercept [rate-interval minutes]
[burst-rate attacks_per_sec] [average-rate attacks_per_sec]
```

Example:

```
hostname(config)# threat-detection statistics tcp-intercept rate-interval 60 burst-rate
800 average-rate 600
```

The **rate-interval** keyword sets the size of the history monitoring window, between 1 and 1440 minutes. The default is 30 minutes. During this interval, the ASA samples the number of attacks 30 times.

The **burst-rate** keyword sets the threshold for syslog message generation, between 25 and 2147483647. The default is 400 per second. When the burst rate is exceeded, syslog message 733104 is generated.

The **average-rate** keyword sets the average rate threshold for syslog message generation, between 25 and 2147483647. The default is 200 per second. When the average rate is exceeded, syslog message 733105 is generated.

Note This command is available in multiple context mode, unlike the other threat-detection commands.

Configure Scanning Threat Detection

You can configure scanning threat detection to identify attackers and optionally shun them.

Procedure

Step 1 Enable scanning threat detection.

threat-detection scanning-threat [**shun** [**except** {**ip-address** *ip_address mask* | **object-group** *network_object_group_id*}]]

Example:

```
hostname(config)# threat-detection scanning-threat shun except ip-address 10.1.1.0
255.255.255.0
```

By default, the system log message 733101 is generated when a host is identified as an attacker. Enter this command multiple times to identify multiple IP addresses or network object groups to exempt from shunning.

Step 2 (Optional) Set the duration of the shun for attacking hosts.

threat-detection scanning-threat shun duration *seconds*

Example:

```
hostname(config)# threat-detection scanning-threat shun duration 2000
```

Step 3 (Optional) Change the default event limit for when the ASA identifies a host as an attacker or as a target.

threat-detection rate scanning-threat rate-interval *rate_interval* **average-rate** *av_rate* **burst-rate** *burst_rate*

Example:

```
hostname(config)# threat-detection rate scanning-threat rate-interval 1200 average-rate 10
burst-rate 20

hostname(config)# threat-detection rate scanning-threat rate-interval 2400 average-rate 10
burst-rate 20
```

If you already configured this command as part of the basic threat detection configuration, then those settings are shared with the scanning threat detection feature; you cannot configure separate rates for basic and scanning threat detection. If you do not set the rates using this command, the default values are used for both the scanning threat detection feature and the basic threat detection feature. You can configure up to three different rate intervals, by entering separate commands.

Monitoring Threat Detection

The following topics explain how to monitor threat detection and view traffic statistics.

Monitoring Basic Threat Detection Statistics

To display basic threat detection statistics, use the following command:

```
show threat-detection rate [min-display-rate min_display_rate]
[acl-drop | bad-packet-drop | conn-limit-drop | dos-drop | fw-drop |
icmp-drop | inspect-drop | interface-drop | scanning-threat | syn-attack]
```

The **min-display-rate** *min_display_rate* argument limits the display to statistics that exceed the minimum display rate in events per second. You can set the *min_display_rate* between 0 and 2147483647.

The other arguments let you limit the display to specific categories. For a description of each event type, see Basic Threat Detection Statistics, page 18-2.

The output shows the average rate in events/sec over two fixed time periods: the last 10 minutes and the last 1 hour. It also shows: the current burst rate in events/sec over the last completed burst interval, which is 1/30th of the average rate interval or 10 seconds, whichever is larger; the number of times the rates were exceeded (triggered); and the total number of events over the time periods.

The ASA stores the count at the end of each burst period, for a total of 30 completed burst intervals. The unfinished burst interval presently occurring is not included in the average rate. For example, if the average rate interval is 20 minutes, then the burst interval is 20 seconds. If the last burst interval was from 3:00:00 to 3:00:20, and you use the **show** command at 3:00:25, then the last 5 seconds are not included in the output.

The only exception to this rule is if the number of events in the unfinished burst interval already exceeds the number of events in the oldest burst interval (#1 of 30) when calculating the total events. In that case, the ASA calculates the total events as the last 29 complete intervals, plus the events so far in the unfinished burst interval. This exception lets you monitor a large increase in events in real time.

You can clear statistics using the **clear threat-detection rate** command.

The following is sample output from the **show threat-detection rate** command:

```
hostname# show threat-detection rate

                        Average(eps)    Current(eps) Trigger         Total events
  10-min ACL  drop:                0               0       0                   16
```

```
1-hour ACL  drop:                0          0        0              112
1-hour SYN attck:                5          0        2            21438
10-min  Scanning:                0          0       29              193
1-hour  Scanning:              106          0       10           384776
1-hour Bad  pkts:               76          0        2           274690
10-min  Firewall:                0          0        3               22
1-hour  Firewall:               76          0        2           274844
10-min DoS attck:                0          0        0                6
1-hour DoS attck:                0          0        0               42
10-min Interface:                0          0        0              204
1-hour Interface:               88          0        0           318225
```

Monitoring Advanced Threat Detection Statistics

To monitor advanced threat detection statistics, use the commands shown in the following table. The display output shows the following:

- The average rate in events/sec over fixed time periods.
- The current burst rate in events/sec over the last completed burst interval, which is 1/30th of the average rate interval or 10 seconds, whichever is larger
- The number of times the rates were exceeded (for dropped traffic statistics only)
- The total number of events over the fixed time periods.

The ASA stores the count at the end of each burst period, for a total of 30 completed burst intervals. The unfinished burst interval presently occurring is not included in the average rate. For example, if the average rate interval is 20 minutes, then the burst interval is 20 seconds. If the last burst interval was from 3:00:00 to 3:00:20, and you use the **show** command at 3:00:25, then the last 5 seconds are not included in the output.

The only exception to this rule is if the number of events in the unfinished burst interval already exceeds the number of events in the oldest burst interval (#1 of 30) when calculating the total events. In that case, the ASA calculates the total events as the last 29 complete intervals, plus the events so far in the unfinished burst interval. This exception lets you monitor a large increase in events in real time.

Command	Purpose
show threat-detection statistics [**min-display-rate** *min_display_rate*] **top** [[**access-list** | **host** | **port-protocol**] [**rate-1** | **rate-2** | **rate-3**] | **tcp-intercept** [**all**] **detail**]]	Displays the top 10 statistics. If you do not enter any options, the top 10 statistics are shown for all categories. The **min-display-rate** *min_display_rate* argument limits the display to statistics that exceed the minimum display rate in events per second. You can set the *min_display_rate* between 0 and 2147483647. Following rows explain optional keywords.

Command	Purpose
show threat-detection statistics [**min-display-rate** *min_display_rate*] **top access-list** [**rate-1** | **rate-2** | **rate-3**]	To view the top 10 ACEs that match packets, including both permit and deny ACEs, use the **access-list** keyword. Permitted and denied traffic are not differentiated in this display. If you enable basic threat detection using the **threat-detection basic-threat** command, you can track ACL denies using the **show threat-detection rate acl-drop** command. The **rate-1** keyword shows the statistics for the smallest fixed rate intervals available in the display; **rate-2** shows the next largest rate interval; and **rate-3**, if you have three intervals defined, shows the largest rate interval. For example, the display shows statistics for the last 1 hour, 8 hours, and 24 hours. If you set the **rate-1** keyword, the ASA shows only the 1 hour time interval.
show threat-detection statistics [**min-display-rate** *min_display_rate*] **top host** [**rate-1** | **rate-2** | **rate-3**]	To view only host statistics, use the **host** keyword. **Note**: Due to the threat detection algorithm, an interface used as a combination failover and state link could appear in the top 10 hosts; this is expected behavior, and you can ignore this IP address in the display.
show threat-detection statistics [**min-display-rate** *min_display_rate*] **top port-protocol** [**rate-1** | **rate-2** | **rate-3**]	To view statistics for ports and protocols, use the **port-protocol** keyword. The **port-protocol** keyword shows statistics for both ports and protocols (both must be enabled for the display), and shows the combined statistics of TCP/UDP port and IP protocol types. TCP (protocol 6) and UDP (protocol 17) are not included in the display for IP protocols; TCP and UDP ports are, however, included in the display for ports. If you only enable statistics for one of these types, port or protocol, then you will only view the enabled statistics.
show threat-detection statistics [**min-display-rate** *min_display_rate*] **top tcp-intercept** [**all**] **detail**]]	To view TCP Intercept statistics, use the **tcp-intercept** keyword. The display includes the top 10 protected servers under attack. The **all** keyword shows the history data of all the traced servers. The **detail** keyword shows history sampling data. The ASA samples the number of attacks 30 times during the rate interval, so for the default 30 minute period, statistics are collected every 60 seconds.
show threat-detection statistics [**min-display-rate** *min_display_rate*] **host** [*ip_address* [*mask*]]	Displays statistics for all hosts or for a specific host or subnet.
show threat-detection statistics [**min-display-rate** *min_display_rate*] **port** [*start_port*[**-***end_port*]]	Displays statistics for all ports or for a specific port or range of ports.
show threat-detection statistics [**min-display-rate** *min_display_rate*] **protocol** [*protocol_number* | **ah** | **eigrp** | **esp** | **gre** | **icmp** | **icmp6** | **igmp** | **igrp** | **ip** | **ipinip** | **ipsec** | **nos** | **ospf** | **pcp** | **pim** | **pptp** | **snp** | **tcp** | **udp**]	Displays statistics for all IP protocols or for a specific protocol. The *protocol_number* argument is an integer between 0 and 255.

Evaluating Host Threat Detection Statistics

The following is sample output from the **show threat-detection statistics host** command:

```
hostname# show threat-detection statistics host

                        Average(eps)    Current(eps) Trigger         Total events
Host:10.0.0.1: tot-ses:289235 act-ses:22571 fw-drop:0 insp-drop:0 null-ses:21438 bad-acc:0
  1-hour Sent byte:              2938               0       0             10580308
```

```
  8-hour Sent byte:                367              0       0              10580308
 24-hour Sent byte:                122              0       0              10580308
  1-hour Sent pkts:                 28              0       0                104043
  8-hour Sent pkts:                  3              0       0                104043
 24-hour Sent pkts:                  1              0       0                104043
  20-min Sent drop:                  9              0       1                 10851
  1-hour Sent drop:                  3              0       1                 10851
  1-hour Recv byte:               2697              0       0               9712670
  8-hour Recv byte:                337              0       0               9712670
 24-hour Recv byte:                112              0       0               9712670
  1-hour Recv pkts:                 29              0       0                104846
  8-hour Recv pkts:                  3              0       0                104846
 24-hour Recv pkts:                  1              0       0                104846
  20-min Recv drop:                 42              0       3                 50567
  1-hour Recv drop:                 14              0       1                 50567
Host:10.0.0.0: tot-ses:1 act-ses:0 fw-drop:0 insp-drop:0 null-ses:0 bad-acc:0
  1-hour Sent byte:                  0              0       0                   614
  8-hour Sent byte:                  0              0       0                   614
 24-hour Sent byte:                  0              0       0                   614
  1-hour Sent pkts:                  0              0       0                     6
  8-hour Sent pkts:                  0              0       0                     6
 24-hour Sent pkts:                  0              0       0                     6
  20-min Sent drop:                  0              0       0                     4
  1-hour Sent drop:                  0              0       0                     4
  1-hour Recv byte:                  0              0       0                   706
  8-hour Recv byte:                  0              0       0                   706
 24-hour Recv byte:                  0              0       0                   706
  1-hour Recv pkts:                  0              0       0                     7
```

The following table explains the output.

Table 18-3 show threat-detection statistics host

Field	Description
Host	The host IP address.
tot-ses	The total number of sessions for this host since it was added to the database.
act-ses	The total number of active sessions that the host is currently involved in.
fw-drop	The number of firewall drops. Firewall drops is a combined rate that includes all firewall-related packet drops tracked in basic threat detection, including ACL denials, bad packets, exceeded connection limits, DoS attack packets, suspicious ICMP packets, TCP SYN attack packets, and UDP session with no return data attack packets. It does not include non-firewall-related drops such as interface overload, packets failed at application inspection, and scanning attack detected.
insp-drop	The number of packets dropped because they failed application inspection.
null-ses	The number of null sessions, which are TCP SYN sessions that did not complete within the 3-second timeout, and UDP sessions that did not have any data sent by its server 3 seconds after the session starts.
bad-acc	The number of bad access attempts to host ports that are in a closed state. When a port is determined to be in a null session (see the null-ses field description), the port state of the host is set to HOST_PORT_CLOSE. Any client accessing the port of the host is immediately classified as a bad access without the need to wait for a timeout.

Table 18-3 show threat-detection statistics host (continued)

Field	Description
Average(eps)	The average rate in events/sec over each time period. The ASA stores the count at the end of each burst period, for a total of 30 completed burst intervals. The unfinished burst interval presently occurring is not included in the average rate. For example, if the average rate interval is 20 minutes, then the burst interval is 20 seconds. If the last burst interval was from 3:00:00 to 3:00:20, and you use the **show** command at 3:00:25, then the last 5 seconds are not included in the output. The only exception to this rule is if the number of events in the unfinished burst interval already exceeds the number of events in the oldest burst interval (#1 of 30) when calculating the total events. In that case, the ASA calculates the total events as the last 29 complete intervals, plus the events so far in the unfinished burst interval. This exception lets you monitor a large increase in events in real time.
Current(eps)	The current burst rate in events/sec over the last completed burst interval, which is 1/30th of the average rate interval or 10 seconds, whichever is larger. For the example specified in the Average(eps) description, the current rate is the rate from 3:19:30 to 3:20:00
Trigger	The number of times the dropped packet rate limits were exceeded. For valid traffic identified in the sent and received bytes and packets rows, this value is always 0, because there are no rate limits to trigger for valid traffic.
Total events	The total number of events over each rate interval. The unfinished burst interval presently occurring is not included in the total events. The only exception to this rule is if the number of events in the unfinished burst interval already exceeds the number of events in the oldest burst interval (#1 of 30) when calculating the total events. In that case, the ASA calculates the total events as the last 29 complete intervals, plus the events so far in the unfinished burst interval. This exception lets you monitor a large increase in events in real time.
20-min, 1-hour, 8-hour, and 24-hour	Statistics for these fixed rate intervals. For each interval: • Sent byte—The number of successful bytes sent from the host. • Sent pkts—The number of successful packets sent from the host. • Sent drop—The number of packets sent from the host that were dropped because they were part of a scanning attack. • Recv byte—The number of successful bytes received by the host. • Recv pkts—The number of successful packets received by the host. • Recv drop—the number of packets received by the host that were dropped because they were part of a scanning attack.

Monitoring Shunned Hosts, Attackers, and Targets

To monitor and manage shunned hosts and attackers and targets, use the following commands:

- **show threat-detection shun**

  Displays the hosts that are currently shunned. For example:

```
hostname# show threat-detection shun
Shunned Host List:
10.1.1.6
192.168.6.7
```

- **clear threat-detection shun** [*ip_address* [*mask*]]

  Releases a host from being shunned. If you do not specify an IP address, all hosts are cleared from the shun list.

  For example, to release the host at 10.1.1.6, enter the following command:

```
hostname# clear threat-detection shun 10.1.1.6
```

- **show threat-detection scanning-threat** [**attacker** | **target**]

  Displays hosts that the ASA decides are attackers (including hosts on the shun list), and displays the hosts that are the target of an attack. If you do not enter an option, both attackers and target hosts are displayed. For example:

```
hostname# show threat-detection scanning-threat attacker
10.1.2.3
10.8.3.6
209.165.200.225
```

Examples for Threat Detection

The following example configures basic threat detection statistics, and changes the DoS attack rate settings. All advanced threat detection statistics are enabled, with the host statistics number of rate intervals lowered to 2. The TCP Intercept rate interval is also customized. Scanning threat detection is enabled with automatic shunning for all addresses except 10.1.1.0/24. The scanning threat rate intervals are customized.

```
threat-detection basic-threat
threat-detection rate dos-drop rate-interval 600 average-rate 60 burst-rate 100
threat-detection statistics
threat-detection statistics host number-of-rate 2
threat-detection statistics tcp-intercept rate-interval 60 burst-rate 800 average-rate 600
threat-detection scanning-threat shun except ip-address 10.1.1.0 255.255.255.0
threat-detection rate scanning-threat rate-interval 1200 average-rate 10 burst-rate 20
threat-detection rate scanning-threat rate-interval 2400 average-rate 10 burst-rate 20
```

History for Threat Detection

Feature Name	Platform Releases	Description
Basic and advanced threat detection statistics, scanning threat detection	8.0(2)	Basic and advanced threat detection statistics, scanning threat detection was introduced. The following commands were introduced: **threat-detection basic-threat**, **threat-detection rate**, **show threat-detection rate**, **clear threat-detection rate,** **threat-detection statistics**, **show threat-detection statistics**, **threat-detection scanning-threat**, **threat-detection rate scanning-threat**, **show threat-detection scanning-threat**, **show threat-detection shun**, **clear threat-detection shun**.
Shun duration	8.0(4)/8.1(2)	You can now set the shun duration, The following command was introduced: **threat-detection scanning-threat shun duration**.
TCP Intercept statistics	8.0(4)/8.1(2)	TCP Intercept statistics were introduced. The following commands were modified or introduced: **threat-detection statistics tcp-intercept**, **show threat-detection statistics top tcp-intercept**, **clear threat-detection statistics**.
Customize host statistics rate intervals	8.1(2)	You can now customize the number of rate intervals for which statistics are collected. The default number of rates was changed from 3 to 1. The following command was modified: **threat-detection statistics host number-of-rates**.
Burst rate interval changed to 1/30th of the average rate.	8.2(1)	In earlier releases, the burst rate interval was 1/60th of the average rate. To maximize memory usage, the sampling interval was reduced to 30 times during the average rate.
Customize port and protocol statistics rate intervals	8.3(1)	You can now customize the number of rate intervals for which statistics are collected. The default number of rates was changed from 3 to 1. The following commands were modified: **threat-detection statistics port number-of-rates**, **threat-detection statistics protocol number-of-rates**.
Improved memory usage	8.3(1)	The memory usage for threat detection was improved. The following command was introduced: **show threat-detection memory**.