Zone-Based Policy Firewalls

This module describes the Cisco unidirectional firewall policy between groups of interfaces known as zones. Prior to the release of the Cisco unidirectional firewall policy, Cisco firewalls were configured only as an inspect rule on interfaces. Traffic entering or leaving the configured interface was inspected based on the direction in which the inspect rule was applied.

Note Cisco IOS XE supports Virtual Fragmentation Reassembly (VFR) on zone-based firewall configuration. When you enable the firewall on an interface by adding the interface to a zone, VFR is configured automatically on the same interface.

- Finding Feature Information, page 1
- Prerequisites for Zone-Based Policy Firewall, page 2
- Restrictions for Zone-Based Policy Firewall, page 2
- Information About Zone-Based Policy Firewalls, page 3
- How to Configure Zone-Based Policy Firewalls, page 16
- Configuration Examples for Zone-Based Policy Firewalls, page 58
- Additional References for Zone-Based Policy Firewalls, page 62
- Feature Information for Zone-Based Policy Firewalls, page 63

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Zone-Based Policy Firewall

- Before you create zones, you must consider what should constitute zones. The general guideline is that you should group interfaces that are similar when they are viewed from a security perspective.
- The Wide Area Application Services (WAAS) and Cisco IOS firewall interoperability capability applies only on the Zone-Based Policy Firewall feature in Cisco IOS Release 12.4(11)T2 and later releases.

Restrictions for Zone-Based Policy Firewall

- If a configuration includes both security zones and inspect rules on interfaces (the old methodology), the configuration may work, but that type of configuration is not recommended.
- In Cisco IOS Releases 12.4(20)T and 12.4(15)T, the cumulative counters in the **show policy-map type inspect zone-pair** command output do not increment for **match** statements in a nested class-map configuration. The problem with counters exists regardless of whether the top-level class map uses the **match-any** or **match-all** keyword. For more information, see the "Example: Protocol Match Data Not Incrementing for a Class Map" section.
- In Cisco IOS Release 12.4(15)T, if the Simple Mail Transfer Protocol (SMTP) is configured and you need to configure the Extended SMTP (ESMTP), you must configure the **no match protocol smtp** command before configuring the **match protocol smtp extended** command. To revert to regular SMTP inspection, use the **no match protocol smtp extended** command and then enter the **match protocol smtp** command. If these commands are not configured in the proper order, the following error is displayed:
```
%Cannot add this filter. Remove match protocol smtp filter and then add this filter.
```
- In a Wide-Area Application Services (WAAS) and firewall configuration, all packets processed by a Wide Area Application Engine (WAE) must pass through the firewall in both directions to support the Web Cache Coordination Protocol (WCCP). This situation occurs because the Layer 2 redirect is not available in Cisco IOS Release 12.4T. If Layer 2 redirect is configured on the WAE, the system defaults to the generic routing encapsulation (GRE) redirect to continue to function.
- In a WAAS and firewall configuration, WCCP does not support traffic redirection using policy-based routing (PBR).
- The zone-based firewall cannot interoperate with WAAS and WCCP, when WCCP is configured with Layer 2 redirect method. The firewall only supports generic routing encapsulation (GRE) redirection.
- The zone-based firewall does not support when Layer 2 redirect is configured as a redirection method in WAAS. Only GRE as a redirection method is supported.
- Stateful inspection support for multicast traffic is not supported between any zones, including the self zone. Use Control Plane Policing for the protection of the control plane against multicast traffic.
- When an in-to-out zone-based policy is configured to match the Internet Control Message Protocol (ICMP) on a Windows system, the **traceroute** command works. However, the same configuration on an Apple system does not work because it uses a UDP-based traceroute. To overcome this issue, configure an out-to-in zone-based policy with the **icmp time-exceeded** and **icmp host unreachable** commands with the **pass** command (not the **inspect** command).
- A UDP-based traceroute is not supported through Internet Control Message Protocol (ICMP) inspection.

- To allow GRE and Encapsulating Security Payload (ESP) protocol traffic through a zone-based policy firewall, use the **pass** command. The GRE and the ESP protocols do not support stateful inspection and if you use the **inspect** command, the traffic for these protocols is dropped.
- In Cisco IOS Release 15.3(1)T and later releases, the peer-to-peer protocols are deprecated. You cannot configure the peer-to-peer protocols with zone-based policy firewalls.
- The zone-based firewall supports only Skinny Client Control Protocol (SCCP) protocol versions up to 17. SCCP versions above 17 are not tested or supported. If you are using an SCCP version that is above 17, either use the **pass** command instead of the **inspect** command or allow the out-to-in traffic through access control lists (ACLs).
- Configuring zone-based policy firewall high availability with Network Address Translation (NAT) and NAT high availability with zone-based policy firewalls is not recommended.
- If you have configured multiple class matching for Layer 7 policies, the reset action takes precedence over other actions such as pass and allow. Unlike Layer 4 policies, the zone-based firewall classification runs through all class maps even though the traffic has already matched a class map.

Information About Zone-Based Policy Firewalls

Top-Level Class Maps and Policy Maps

Top-level class maps allow you to identify the traffic stream at a high level. Identifying the traffic stream is accomplished by using the **match access-group** and **match protocol** commands. Top-level class maps are also referred to as Layer 3 and Layer 4 class maps.

Top-level policy maps allow you to define high-level actions by using the **inspect**, **drop**, **pass**, and **urlfilter** keywords. You can attach maps to a target (zone pair).

Note Only inspect type policies can be configured on a zone pair.

With CSCto44113 fix, only Layer 4 policy maps will be inspected by the firewall when you configure the **access-group match** command. Prior to this fix, when the **access-group match** command was configured, both Layer 4 and Layer 7 policy maps were inspected.

Application-Specific Class Maps and Policy Maps

Application-specific class maps allow you to identify traffic based on the attributes of a given protocol. All match conditions in these class maps are specific to an application (for example, HTTP or SMTP). Application-specific class maps are identified by an additional subtype that generally is the protocol name (HTTP or SMTP), in addition to the type **inspect**.

Application-specific policy maps are used to specify a policy for an application protocol. For example, if you want to drop HTTP traffic with Unique Resource Identifier (URI) lengths exceeding 256 bytes, you must configure an HTTP policy map. Application-specific policy maps cannot be attached directly to a target (zone pair). They must be configured as “child” policies in a top-level Layer 3 or Layer 4 policy map.

Overview of Zones

A zone is a group of interfaces that have similar functions or features. Zones provide a way to specify where a Cisco firewall is applied.

For example, on a device, Gigabit Ethernet interface 0/0/0 and Gigabit Ethernet interface 0/0/1 may be connected to the local LAN. These two interfaces are similar because they represent the internal network, so they can be grouped into a zone for firewall configurations.

By default, the traffic between interfaces in the same zone is not subject to any policy and passes freely. Firewall zones are used for security features.

Note Zones may not span interfaces in different VPN routing and forwarding (VRF) instances.

When a zone-based policy firewall is enabled for TCP keepalive traffic and the host behind the firewall is undergoing an ungraceful disconnect, TCP keepalive works only when the configured TCP timeout is complete. On receiving an out-of-window reset (RST) packet, the firewall sends an empty acknowledge (ACK) packet to the initiator of the RST packet. This ACK has the current sequence (SEQ) and the ACK number from the firewall session. On receiving this ACK, the client sends an RST packet with the SEQ number that is equal to the ACK number in the ACK packet. The firewall processes this RST packet, clears the firewall session, and passes the RST packet.

Security Zones

A security zone is a group of interfaces to which a policy can be applied.

Grouping interfaces into zones involves two procedures:

- Creating a zone so that interfaces can be attached to it.
- Configuring an interface to be a member of a given zone.

By default, traffic flows among interfaces that are members of the same zone.

When an interface is a member of a security zone, all traffic (except traffic going to the device or initiated by the device) between that interface and an interface within a different zone is dropped by default. To permit traffic to and from a zone-member interface and another interface, you must make that zone part of a zone pair and apply a policy to that zone pair. If the policy permits traffic through **inspect** or **pass** actions, traffic can flow through the interface.

The following are basic rules to consider when setting up zones:

- Traffic from a zone interface to a nonzone interface or from a nonzone interface to a zone interface is always dropped; unless default zones are enabled (default zone is a nonzone interface).
- Traffic between two zone interfaces is inspected if there is a zone pair relationship for each zone and if there is a configured policy for that zone pair.
- By default, all traffic between two interfaces in the same zone is always allowed.
- A zone pair can be configured with a zone as both source and destination zones. An inspect policy can be configured on this zone pair to inspect or drop the traffic between two interfaces in the same zone.
- An interface cannot be part of a zone and a legacy inspect policy at the same time.

- An interface can be a member of only one security zone.
- When an interface is a member of a security zone, all traffic to and from that interface is blocked unless you configure an explicit interzone policy on a zone pair involving that zone.
- Traffic cannot flow between an interface that is a member of a security zone and an interface that is not a member of a security zone because a policy can be applied only between two zones.
- For traffic to flow among all interfaces in a device, all interfaces must be members of one security zone or another. This is particularly important because after you make an interface a member of a security zone, a policy action (such as **inspect** or **pass**) must explicitly allow packets. Otherwise, packets are dropped.
- If an interface on a device cannot be part of a security zone or firewall policy, you may have to add that interface in a security zone and configure a “pass all” policy (that is, a “dummy” policy) between that zone and other zones to which a traffic flow is desired.
- You cannot apply an access control list (ACL) between security zones or on a zone pair.
- An ACL cannot be applied between security zones and zone pairs. Include the ACL configuration in a class map, and use policy maps to drop traffic.
- An ACL on an interface that is a zone member should not be restrictive (strict).
- All interfaces in a security zone must belong to the same VPN routing and forwarding (VRF) instance.
- You can configure policies between security zones whose member interfaces are in separate VRFs. However, traffic may not flow between these VRFs if the configuration does not allow it.
- If traffic does not flow between VRFs (because route-leaking between VRFs is not configured), the policy across VRFs is not executed. This is a configuration mistake on the routing side, not on the policy side.
- Traffic between interfaces in the same security zone is not subject to any policy; traffic passes freely.
- Source and destination zones in a zone pair must be of the type security.
- The same zone cannot be defined as both source and destination zones.

A policy is applied to an initiating packet of a traffic flow. After the initial packet has been classified and permitted, traffic flows between peers with no further reclassification of the packet (this means that bidirectional traffic flow is allowed after the initial classification). If you have a zone pair between Zone Z1 and Zone Z2, and no zone pair between Zone Z2 and Zone Z1, all traffic that is initiated from Zone Z2 is blocked. Traffic from Zone Z1 to Zone Z2 is permitted or denied based on the zone pair policy.

For traffic to flow among all interfaces in a device, all interfaces must be members of security zones or the default zone.

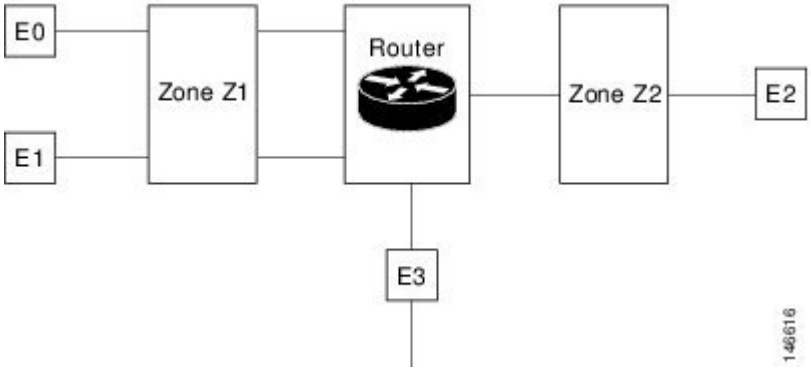
It is not necessary for all device interfaces to be members of security zones.

The figure below illustrates the following:

- Interfaces E0 and E1 are members of security zone Z1.
- Interface E2 is a member of security zone Z2.

- Interface E3 is not a member of any security zone.

Figure 1: Security Zone Restrictions



The following situations exist:

- The zone pair and policy are configured in the same zone. If no policy is configured for Z1 and Z2, traffic will flow freely between E0 and E1, but not between E0 or E1 to E2. A zone pair and policy may be created to inspect this traffic.
- If no policies are configured, traffic will not flow between any other interfaces (for example, E0 and E2, E1 and E2, E3 and E1, and E3 and E2).
- Traffic can flow between E0 or E1 and E2 only when an explicit policy permitting traffic is configured between zone Z1 and zone Z2.
- Traffic can never flow between E3 and E0, E1, or E2 unless default zones are enabled and a zone pair is created between the default zone and other zones.

Virtual Interfaces as Members of Security Zones

A virtual template interface is a logical interface configured with generic configuration information for a specific purpose or for a configuration common to specific users, plus device-dependent information. The template contains Cisco software interface commands that are applied to virtual access interfaces. To configure a virtual template interface, use the **interface virtual-template** command.

Zone member information is acquired from a RADIUS server and the dynamically created interface is made a member of that zone.

The **zone-member security** command adds the dynamic interface to the corresponding zone.

Zone Pairs

A zone pair allows you to specify a unidirectional firewall policy between two security zones.

To define a zone pair, use the **zone-pair security** command. The direction of the traffic is specified by source and destination zones. The source and destination zones of a zone pair must be security zones.

You can select the default or self zone as either the source or the destination zone. The self zone is a system-defined zone which does not have any interfaces as members. A zone pair that includes the self zone,

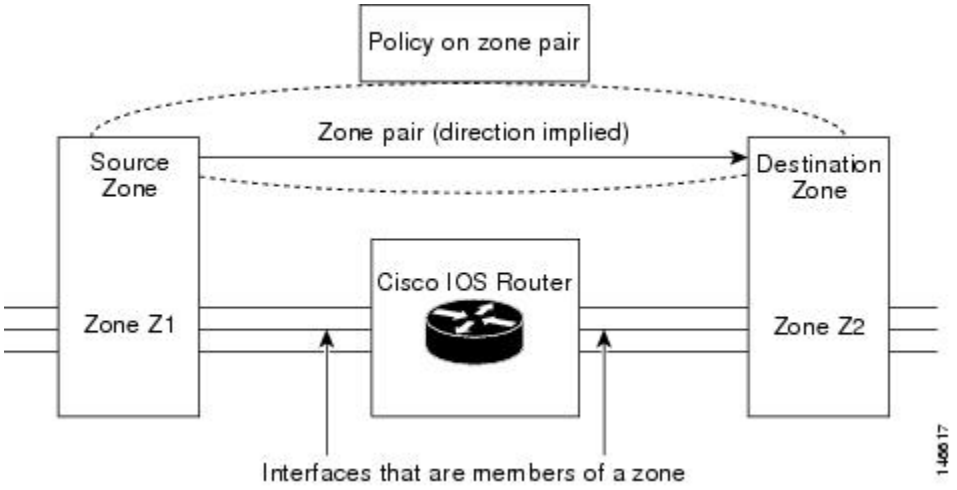
along with the associated policy, applies to traffic directed to the device or traffic generated by the device. It does not apply to traffic through the device.

The most common usage of firewall is to apply them to traffic through a device, so you need at least two zones (that is, you cannot use the self zone).

To permit traffic between zone member interfaces, you must configure a policy permitting (or inspecting) traffic between that zone and another zone. To attach a firewall policy map to the target zone pair, use the **service-policy type inspect** command.

The figure below shows the application of a firewall policy to traffic flowing from zone Z1 to zone Z2, which means that the ingress interface for the traffic is a member of zone Z1 and the egress interface is a member of zone Z2.

Figure 2: Zone Pairs



If there are two zones and you require policies for traffic going in both directions (from Z1 to Z2 and Z2 to Z1), you must configure two zone pairs (one for each direction).

If a policy is not configured between zone pairs, traffic is dropped. However, it is not necessary to configure a zone pair and a service policy solely for the return traffic. By default, return traffic is not allowed. If a service policy inspects the traffic in the forward direction and there is no zone pair and service policy for the return traffic, the return traffic is inspected. If a service policy passes the traffic in the forward direction and there is no zone pair and service policy for the return traffic, the return traffic is dropped. In both these cases, you need to configure a zone pair and a service policy to allow the return traffic. In the above figure, it is not mandatory that you configure a zone pair source and destination for allowing return traffic from Z2 to Z1. The service policy on Z1 to Z2 zone pair takes care of it.

A zone-based firewall drops a packet if it is not explicitly allowed by a rule or policy in contrast to a legacy firewall, which permits a packet if it is not explicitly denied by a rule or policy by default.

A zone-based firewall behaves differently when handling intermittent Internet Control Message Protocol (ICMP) responses generated within a zone because of the traffic flowing between in-zones and out-zones.

In a configuration where an explicit policy is configured for the self zone to go out of its zone and for the traffic moving between the in-zone and out-zone, if any intermittent ICMP responses are generated, then the zone-based firewall looks for an explicit permit rule for the ICMP in the self zone to go out of its zone. An explicit inspect rule for the ICMP for the self zone to go out-zone may not help because there is no session associated with the intermittent ICMP responses.

Zones and Inspection

Zone-based policy firewalls examine source and destination zones from the ingress and egress interfaces for a firewall policy. It is not necessary that all traffic flowing to or from an interface be inspected; you can designate that individual flows in a zone pair be inspected through your policy map that you apply across the zone pair. The policy map will contain class maps that specify individual flows. Traffic with the inspect action will create a connection in the firewall table and be subject to state checking. Traffic with the pass action will bypass the zone firewall completely, not creating any sessions.

You can also configure **inspect** parameters like TCP thresholds and timeouts on a per-flow basis.

Zones and ACLs

Access control lists (ACLs) applied to interfaces that are members of zones are processed before the policy is applied on the zone pair. You must ensure that interface ACLs do not interfere with the policy firewall traffic when there are policies between zones.

Pinholes (ports opened through a firewall that allows applications-controlled access to a protected network) are not punched for return traffic in interface ACLs.

Zones and VRF-Aware Firewalls

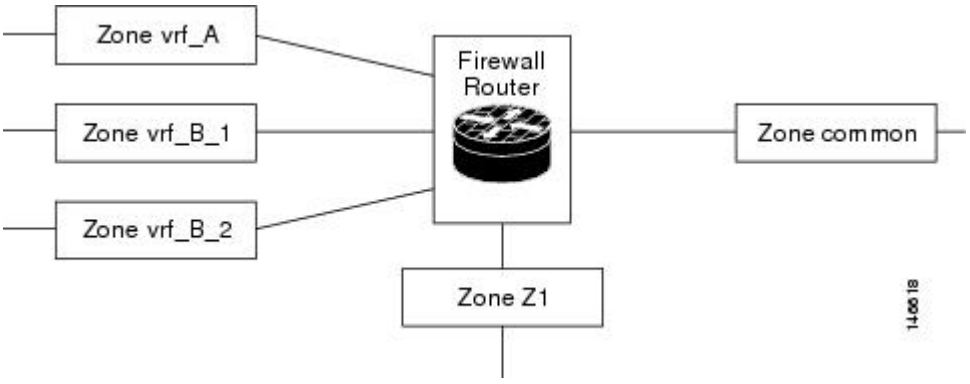
The Cisco firewall is VPN routing and forwarding (VRF)-aware. It handles IP address overlap across different VRFs, separate thresholds, and timeouts for VRFs. All interfaces in a zone must belong to the same VRF.

However, you should not group interfaces from different VRFs in the same zone because VRFs belong to different entities that typically have their own policies.

You can configure a zone pair between two zones that contain different VRFs, as shown in the figure below.

When multiple VRFs are configured on a device and an interface provides common services to all the VRFs (for example, Internet service), you should place that interface in a separate zone. You can then define policies between the common zone and other zones. (There can be one or more zones per VRF.)

Figure 3: Zones and VRF



In the figure above, the interface providing common services is a member of the zone “common.” All of VRF A is in a single zone, vrf_A. VRF B, which has multiple interfaces, is partitioned into multiple zones vrf_B_1

and vrf_B_2. Zone Z1 does not have VRF interfaces. You can specify policies between each of these zones and the common zone. Additionally, you can specify polices between each of the zones vrf_A, vrf_B_n, and Z1 if VRF route export is configured and the traffic patterns make sense. You can configure a policy between zones vrf_A and vrf_B_1, but make sure that traffic can flow between them.

You do not need to specify the global thresholds and timers on a per-VRF basis. Instead, parameters are supplied to the **inspect** action through a parameter map.

Zones and Transparent Firewalls

The Cisco firewall supports transparent firewalls where the interfaces are placed in bridging mode and the firewall inspects the bridged traffic.

To configure a transparent firewall, use the **bridge** command to enable the bridging of a specified protocol in a specified bridge and the **zone-member security** command to attach an interface to a zone. The **bridge** command on the interface indicates that the interface is in bridging mode.

A bridged interface can be a zone member. In a typical case, the Layer 2 domain is partitioned into zones and a policy is applied the same way as for Layer 3 interfaces.

Transparent Firewall Restriction for P2P Inspection

The Cisco firewall uses network-based application recognition (NBAR) for peer-to-peer (P2P) protocol classification and policy enforcement. NBAR is not available for bridged packets; thus, P2P packet inspection is not supported for firewalls with transparent bridging.

Overview of Security Zone Firewall Policies

A class is a way of identifying a set of packets based on its contents. Normally, you define a class so that you can apply an action on the identified traffic that reflects a policy. A class is designated through class maps.

An action is a specific functionality that is typically associated with a traffic class. For example, **inspect**, **drop**, and **pass** are actions.

To create security zone firewall policies, you should complete the following tasks:

- Define a match criterion (class map).
- Associate actions to the match criterion (policy map).
- Attach the policy map to a zone pair (service policy).

The **class-map** command creates a class map to be used for matching packets to a specified class. Packets arriving at the targets (such as the input interface, output interface, or zone pair), that are determined by how the **service-policy** command is configured, are checked against match criteria configured for a class map to determine if the packet belongs to that class.

The **policy-map** command creates or modifies a policy map that can be attached to one or more targets to specify a service policy. Use the **policy-map** command to specify the name of the policy map to be created, added to, or modified before you can configure policies for classes whose match criteria are defined in a class map.

To log firewall drop messages, enable the **drop-log** command under the class-default class in the policy map. For example, consider the following policy map:

```
policy-map type inspect in-out-pol
 class type inspect in-out
  inspect
 class class-default
  drop-log
policy-map type inspect out-in-pol
 class type inspect out-in
  inspect
 class class-default
  drop-log
```

To log dropped packets for an inspect parameter map, use the **log dropped-packets enable** command. The following example shows how to configure logging of dropped packets due to an inspect policy:

```
parameter-map type inspect global
 log dropped-packets enable
```

Class Maps and Policy Maps for Zone-Based Policy Firewalls

Quality of service (QoS) class maps have numerous match criteria; firewalls have fewer match criteria. Firewall class maps are of type inspect and this information controls what shows up under firewall class maps.

A policy is an association of traffic classes and actions. It specifies what actions should be performed on defined traffic classes. An action is a specific function, and it is typically associated with a traffic class. For example, **inspect** and **drop** are actions.

Layer 3 and Layer 4 Class Maps and Policy Maps

Layer 3 and Layer 4 class maps identify traffic streams on which different actions should be performed.

A Layer 3 or Layer 4 policy map is sufficient for the basic inspection of traffic.

The following example shows how to configure class map c1 with the match criteria of ACL 101 and the HTTP protocol, and create an inspect policy map named p1 to specify that packets will be dropped on the traffic at c1:

```
Device(config)# class-map type inspect match-all c1
Device(config-cmap)# match access-group 101
Device(config-cmap)# match protocol http
Device(config)# policy-map type inspect p1
Device(config-pmap)# class type inspect c1
Device(config-pmap-c)# drop
```

Class-Map Configuration Restriction

If traffic meets multiple match criteria, these match criteria must be applied in the order of specific to less specific. For example, consider the following class map:

```
class-map type inspect match-any my-test-cmap
 match protocol http
 match protocol tcp
```

In this example, HTTP traffic must first encounter the **match protocol http** command to ensure that the traffic is handled by the service-specific capabilities of HTTP inspection. If the “match” lines are reversed, and the traffic encounters the **match protocol tcp** command before it is compared to the **match protocol http** command, the traffic will be classified as TCP traffic and inspected according to the capabilities of the TCP inspection component of the firewall. If match protocol TCP is configured first, it will create issues for services

such as FTP and TFTP and for multimedia and voice signaling services such as H.323, Real Time Streaming Protocol (RTSP), Session Initiation Protocol (SIP), and Skinny. These services require additional inspection capabilities to recognize more complex activities.

Rate Limiting (Policing) Traffic Within a Layer 3 and Layer 4 Policy Map

Depending on your releases, you can use the **police** command within an inspect policy to limit the number of concurrent connections allowed for applications such as Instant Messenger (IM) and peer-to-peer (P2P).

To use the **police** command, you must enable Cisco stateful packet inspection within the inspect policy map. If you configure the **police** command without configuring the **inspect** command, you will receive an error message and the **police** command will be rejected.

Compatibility with Existing Police Actions

Police actions provisioned in a modular QoS CLI (MQC) policy map are applied as input and output policies on an interface. An inspect policy map can be applied only to a zone pair and not to an interface. The police action is enforced on traffic that traverses the zone pair. (The direction of the traffic is inherent to the specification of the zone pair.) Thus, a quality of service (QoS) policy that contains a police action can be present on interfaces that make up a zone pair and in an inspect policy map applied across the zone pair. If both police actions are configured, the zone pair police action is executed after the input interface police action, but before the output interface police action. There is no interaction between QoS and the inspect police actions.

Police Restrictions

- The police action is not allowed in policies that are attached to zone pairs that involves a “self” zone. Use Control Plane Policing to perform this task.
- Policing can be specified only in Layer 3 and Layer 4 policy maps; it cannot be specified in Layer 7 policy maps.

Layer 7 Class Maps and Policy Maps

Layer 7 class maps can be used in inspect policy maps only for deep packet inspection (DPI). The DPI functionality is delivered through Layer 7 class maps and policy maps.

To create a Layer 7 class map, use the **class-map type inspect** command for the desired protocol. For example, for the HTTP protocol, enter the **class-map type inspect http** command.

The type of class map (for example, HTTP) determines the match criteria that you can use. If you want to specify HTTP traffic that contains Java applets, you must specify a “match response body java” statement in the context of an “inspect HTTP” class map.

A Layer 7 policy map provides application level inspection of traffic. The policy map can include class maps of the same type.

To create a Layer 7 policy map, specify the protocol in the **policy-map type inspect** command. For example, to create a Layer 7 HTTP policy map, use the **policy-map type inspect http** *policy-map-name* command. Enter the name of the HTTP policy-map for the *policy-map-name* argument.

If you do not specify a protocol name (for example, if you use the **policy-map type inspect** command), you will create a Layer 3 or Layer 4 policy map, which can only be an inspect type policy map.

A Layer 7 policy map must be contained in a Layer 3 or Layer 4 policy map; it cannot be attached directly to a target. To attach a Layer 7 policy map to a top-level policy map, use the **service-policy** command and specify the application name (that is, HTTP, Internet Message Access Protocol [IMAP], Post Office Protocol, version 3 [POP3], Simple Mail Transfer Protocol [SMTP], or SUN Remote Procedure Call [SUNRPC]). The parent class for a Layer 7 policy should have an explicit match criterion that matches only one Layer 7 protocol before the policy is attached.

If the Layer 7 policy map is in a lower level, you must specify the **inspect** action at the parent level for a Layer 7 policy map.

If you have configured multiple classes matching for Layer 7 policies, the reset action takes precedence over other actions such as pass and allow. Unlike Layer 4 policies, the zone-based firewall classification runs through all class maps even though the traffic has already matched a class map.

In the following example, policy map p1 has two classes, c1 and c2 attached to it. However, if the traffic matches both c1 and c2, the reset action has precedence over the allow action.

```
Device(config)# policy-map type inspect p1
Device(config-pmap)# class type inspect c1
Device(config-pmap-c)# allow
!
Device(config-pmap)# class type inspect c2
Device(config-pmap-c)# reset
!
```

Layer 7 Supported Protocols

You can create Layer 7 class maps and policy maps for the following protocols:

- America Online (AOL) Instant Messenger (IM) protocol.
- eDonkey peer-to-peer protocol.
- FastTrack traffic peer-to-peer protocol.
- Gnutella Version 2 traffic peer-to-peer protocol.
- H.323 VoIP Protocol Version 4.
- HTTP—Protocol used by web browsers and web servers to transfer files, such as text and graphic files.
- Internet Message Access Protocol (IMAP)—Method of accessing e-mail or bulletin board messages kept on a mail server that is shared.
- I Seek You (ICQ) IM protocol.
- Kazaa Version 2 peer-to-peer protocol.
- MSN Messenger IM protocol.
- Post Office Protocol, Version 3 (POP3)—Protocol that client e-mail applications use to retrieve mail from a mail server.
- SIP—Session Initiation Protocol (SIP).
- SMTP—Simple Network Management Protocol.
- SUNRPC—Sun RPC (Remote Procedure Call).
- Windows Messenger IM Protocol.
- Yahoo IM protocol.

For information on configuring a Layer 7 class map and policy map (policies), see the "Configuring Layer 7 Protocol-Specific Firewall Policies, on page 29" section.

Class-Default Class Map

In addition to user-defined classes, a system-defined class map named class-default represents all packets that do not match any of the user-defined classes in a policy. The class-default class is always the last class in a policy map.

You can define explicit actions for a group of packets that does not match any of the user-defined classes. If you do not configure any actions for the class-default class in an inspect policy, the default action is **drop**.

Note For a class-default in an inspect policy, you can configure only **drop** action or **pass** action.

The following example shows how to use class-default in a policy map. In this example, HTTP traffic is dropped and the remaining traffic is inspected. Class map c1 is defined for HTTP traffic, and class-default is used for a policy map p1.

```
Device(config)# class-map type inspect match-all c1
Device(config-cmap)# match protocol http
Device(config-cmap)# exit
Device(config)# policy-map type inspect p1
Device(config-pmap)# class type inspect c1
Device(config-pmap-c)# drop
Device(config-pmap-c)# exit
Device(config-pmap)# class class-default
Device(config-pmap-c)# drop
```

Hierarchical Policy Maps

A policy can be nested within a policy. A policy that contains a nested policy is called a hierarchical policy.

To create a hierarchical policy, attach a policy directly to a class of traffic. A hierarchical policy contains a child and a parent policy. The child policy is the previously defined policy that is associated with the new policy through the use of the **service-policy** command. The new policy that uses the preexisting policy is the parent policy.

Note There can be a maximum of two levels in a hierarchical inspect service policy.

Parameter Maps

A parameter map allows you to specify parameters that control the behavior of actions and match criteria specified under a policy map and a class map, respectively.

There are two types of parameter maps:

- Inspect parameter map

  An inspect parameter map is optional. If you do not configure a parameter map, the software uses default parameters. Parameters associated with the inspect action apply to all nested actions (if any). If parameters

are specified in both the top and lower levels, parameters in the lower levels override those in the top levels.

- Protocol-specific parameter map

  A parameter map that is required for an Instant Messenger (IM) application (Layer 7) policy map.

Firewall and Network Address Translation

Network Address Translation (NAT) enables private IP internetworks that use nonregistered IP addresses to connect to the Internet. NAT operates on a device, usually connecting two networks, and translates private (not globally unique) addresses in the internal network into legal addresses before packets are forwarded to another network. NAT can be configured to advertise only one address for the entire network to the outside world. A device configured with NAT will have at least one interface to the inside network and one to the outside network.

In a typical environment, NAT is configured at the exit device between a stub domain and the backbone. When a packet leaves the domain, NAT translates the locally significant source address to a global unique address. When a packet enters the domain, NAT translates the globally unique destination address into a local address. If more than one exit point exists, each NAT must have the same translation table. If the software cannot allocate an address because it has run out of addresses, it drops the packet and sends an Internet Control Message Protocol (ICMP) host unreachable packet.

With reference to NAT, the term “inside” refers to those networks that are owned by an organization and that must be translated. Inside this domain, hosts will have addresses in one address space. When NAT is configured and when the hosts are outside, hosts will appear to have addresses in another address space. The inside address space is referred to as the local address space and the outside address space is referred to as the global address space.

Consider a scenario where NAT translates both source and destination IP addresses. A packet is sent to a device from inside NAT with the source address 192.168.1.1 and the destination address 10.1.1.1. NAT translates these addresses and sends the packet to the external network with the source address 209.165.200.225 and the destination address 209.165.200.224.

Similarly, when the response comes back from outside NAT, the source address will be 209.165.200.225 and the destination address will be 209.165.200.224. Therefore, inside NAT, the packets will have a source address of 10.1.1.1 and a destination address of 192.168.1.1.

In this scenario, if you want to create an Application Control Engine (ACE) to be used in a firewall policy, the pre-NAT IP addresses (also known as inside local and outside global addresses) 192.168.1.1 and 209.165.200.224 must be used.

Out-of-Order Packet Processing Support in the Zone-Based Firewall Application

Out-of-Order (OoO) packet processing support for Common Classification Engine (CCE) firewall application and CCE adoptions of the Intrusion Prevention System (IPS) allows packets that arrive out of order to be copied and reassembled in the correct order. The OoO packet processing reduces the need to retransmit dropped packets and reduces the bandwidth needed for the transmission of traffic on a network. To configure OoO support, use the **parameter-map type ooo global** command.

Note IPS sessions use OoO parameters that are configured using the **parameter-map type ooo global** command.

OoO processing is not supported in Simple Mail Transfer Protocol (SMTP) because SMTP supports masking actions that require packet modification.

OoO packet processing support is enabled by default when a Layer 7 policy is configured for Deep Packet Inspection (DPI) for the following protocols:

- AOL IM protocol.
- eDonkey peer-to-peer protocol.
- FastTrack traffic peer-to-peer protocol.
- Gnutella Version 2 traffic peer-to-peer protocol.
- H.323 VoIP Protocol Version 4.
- HTTP—Protocol used by web browsers and web servers to transfer files, such as text and graphic files.
- IMAP—Method of accessing e-mail or bulletin board messages kept on a mail server that is shared.
- ICQ IM Protocol.
- Kazaa Version 2 peer-to-peer protocol.
- Match Protocol SIP—Match Protocol SIP.
- MSN Messenger IM protocol.
- POP3—Protocol that client e-mail applications use to retrieve mail from a mail server.
- SUNRPC—Sun RPC.
- Windows Messenger IM Protocol.
- Yahoo IM protocol.

For information on configuring a Layer 7 class map and policy map (policies), see the "Configuring Layer 7 Protocol-Specific Firewall Policies" section.

Note OoO packets are dropped when IPS and zone-based policy firewall with Layer 4 inspection are enabled.

Intrazone Support in the Zone-Based Firewall Application

Intrazone support allows a zone configuration to include users both inside and outside a network. Intrazone support allows traffic inspection between users belonging to the same zone but different networks. Traffic within the same zone cannot be inspected prior to Cisco IOS Release 15.0(1)M. To configure a zone pair definition with the same zone for source and destination, use the **zone-pair security** command. This allows the functionality of attaching a policy map and inspecting the traffic within the same zone.

How to Configure Zone-Based Policy Firewalls

Configuring Layer 3 and Layer 4 Firewall Policies

Layer 3 and Layer 4 policies are "top-level" policies that are attached to the target (zone pair). Perform the following tasks to configure Layer 3 and Layer 4 firewall policies:

Configuring a Class Map for a Layer 3 and Layer 4 Firewall Policy

Use the following task to configure a class map for classifying network traffic.

Note You must perform at least one match step from Step 4, 5, or 6.

When packets are matched to an access group, a protocol, or a class map, a traffic rate is generated for these packets. In a zone-based firewall policy, only the first packet that creates a session matches the policy. Subsequent packets in this flow do not match the filters in the configured policy, but match the session directly. The statistics related to subsequent packets are shown as part of the inspect action.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map type inspect** [**match-any** | **match-all**] *class-map-name*
4. **match access-group** {*access-group* | **name** *access-group-name*}
5. **match protocol** *protocol-name* [**signature**]
6. **match class-map** *class-map-name*
7. **end**
8. **show policy-map type inspect zone-pair session**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**class-map type inspect** [**match-any** | **match-all**] *class-map-name* **Example:** `Device(config)# class-map type inspect match-all c1`	Creates a Layer 3 or Layer 4 inspect type class map and enters class-map configuration mode.
Step 4	**match access-group** {*access-group* | **name** *access-group-name*} **Example:** `Device(config-cmap)# match access-group 101`	Configures the match criterion for a class map based on the access control list (ACL) name or number.
Step 5	**match protocol** *protocol-name* [**signature**] **Example:** `Device(config-cmap)# match protocol http`	Configures the match criterion for a class map on the basis of a specified protocol. • Only Cisco stateful packet inspection-supported protocols can be used as match criteria in inspect type class maps. • **signature—**Signature-based classification for peer-to-peer packets is enabled.
Step 6	**match class-map** *class-map-name* **Example:** `Device(config-cmap)# match class-map c1`	Specifies a previously defined class as the match criteria for a class map.
Step 7	**end** **Example:** `Device(config-cmap)# end`	Exits class-map configuration mode and returns to privileged EXEC mode.
Step 8	**show policy-map type inspect zone-pair session** **Example:** `Device(config-cmap)# show policy-map type inspect zone-pair session`	(Optional) Displays Cisco stateful packet inspection sessions created because a policy map is applied on the specified zone pair. **Note** The information displayed under the class-map field is the traffic rate (bits per second) of the traffic that belongs to the connection-initiating traffic only. Unless the connection setup rate is significantly high and is sustained for multiple intervals over which the rate is computed, no significant data is shown for the connection.

Creating a Policy Map for a Layer 3 and Layer 4 Firewall Policy

Use this task to create a policy map for a Layer 3 and Layer 4 firewall policy that will be attached to zone pairs.

Note You must perform at least one step from Step 5, 8, 9, or 10.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map type inspect** *policy-map-name*
4. **class type inspect** *class-name*
5. **inspect** [*parameter-map-name*]
6. **drop** [**log**]
7. **pass**
8. **service-policy type inspect** *policy-map-name*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map type inspect** *policy-map-name* **Example:** `Device(config)# policy-map type inspect p1`	Creates a Layer 3 and Layer 4 inspect type policy map and enters policy-map configuration mode.
Step 4	**class type inspect** *class-name* **Example:** `Device(config-pmap)# class type inspect c1`	Specifies the traffic class on which an action to perform and enters policy-map class configuration mode.
Step 5	**inspect** [*parameter-map-name*] **Example:** `Device(config-pmap-c)# inspect inspect-params`	Enables Cisco stateful packet inspection.

	Command or Action	Purpose
Step 6	**drop** [**log**] **Example:** `Device(config-pmap-c)# drop`	(Optional) Drops packets that are matched with the defined class. **Note** Actions **drop** and **pass** are exclusive, and actions **inspect** and **drop** are exclusive; that is, you cannot specify both of them at the same time.
Step 7	**pass** **Example:** `Device(config-pmap-c)# pass`	(Optional) Allows packets that are matched with the defined class.
Step 8	**service-policy type inspect** *policy-map-name* **Example:** `Device(config-pmap-c)# service-policy type inspect p1`	Attaches a firewall policy map to a zone pair.
Step 9	**end** **Example:** `Device(config-pmap-c)# end`	Exits policy-map class configuration mode and returns to privileged EXEC mode.

Configuring a Parameter Map

Depending on your policy, you can configure either an inspect, URL filter, or a protocol-specific parameter map. If you configure a URL filter type or a protocol-specific policy, you must configure a parameter map. However, a parameter map is optional if you are using an inspect type policy.

Note Changes to the parameter map are not reflected on connections already established through the firewall. Changes are applicable only to new connections permitted to the firewall. To ensure that your firewall enforces policies strictly, clear all connections that are allowed in the firewall after you change the parameter map. To clear existing connections, use the **clear zone-pair inspect sessions** command.

Perform one of the following tasks to configure a parameter map:

Creating an Inspect Parameter Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **parameter-map type inspect** {*parameter-map-name* | **global** | **default**}
4. **log** {**dropped-packets** {**disable** | **enable**} | **summary** [**flows** *number*] [**time-interval** *seconds*]}
5. **alert** {**on** | **off**}
6. **audit-trail** {**on** | **off**}
7. **dns-timeout** *seconds*
8. **icmp idle-timeout** *seconds*
9. **max-incomplete** {**low** | **high**} *number-of-connections*
10. **one-minute** {**low** | **high**} *number-of-connections*
11. **sessions maximum** *sessions*
12. **tcp finwait-time** *seconds*
13. **tcp idle-time** *seconds*
14. **tcp max-incomplete host** *threshold* [**block-time** *minutes*]
15. **tcp synwait-time** *seconds*
16. **tcp window-scale-enforcement loose**
17. **udp idle-time** *seconds*
18. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**parameter-map type inspect** {*parameter-map-name* | **global** | **default**} **Example:** `Device(config)# parameter-map type inspect eng-network-profile`	Configures an inspect parameter map for connecting thresholds, timeouts, and other parameters that pertains to the **inspect** action and enters parameter map type inspect configuration mode.
Step 4	**log** {**dropped-packets** {**disable** | **enable**} | **summary** [**flows** *number*] [**time-interval** *seconds*]}	(Optional) Configures packet logging during the firewall activity.

	Command or Action	Purpose
	Example: `Device(config-profile)# log summary flows 15 time-interval 30`	**Note** This command is visible in parameter map type inspect configuration mode only.
Step 5	**alert** {**on** | **off**} **Example:** `Device(config-profile)# alert on`	(Optional) Enables Cisco stateful packet inspection alert messages that are displayed on the console.
Step 6	**audit-trail** {**on** | **off**} **Example:** `Device(config-profile)# audit-trail on`	(Optional) Enables audit trail messages.
Step 7	**dns-timeout** *seconds* **Example:** `Device(config-profile)# dns-timeout 60`	(Optional) Specifies the domain name system (DNS) idle timeout (the length of time for which a DNS lookup session will be managed while there is no activity).
Step 8	**icmp idle-timeout** *seconds* **Example:** `Device(config-profile)# icmp idle-timeout 90`	(Optional) Configures the timeout for Internet Control Message Protocol (ICMP) sessions.
Step 9	**max-incomplete** {**low** | **high**} *number-of-connections* **Example:** `Device(config-profile)# max-incomplete low 800`	(Optional) Defines the number of existing half-open sessions that will cause the Cisco firewall to start and stop deleting half-open sessions.
Step 10	**one-minute** {**low** | **high**} *number-of-connections* **Example:** `Device(config-profile)# one-minute low 300`	(Optional) Defines the number of new unestablished sessions that will cause the system to start deleting half-open sessions and stop deleting half-open sessions.
Step 11	**sessions maximum** *sessions* **Example:** `Device(config-profile)# sessions maximum 200`	(Optional) Sets the maximum number of allowed sessions that can exist on a zone pair. • Use this command to limit the bandwidth used by the sessions.
Step 12	**tcp finwait-time** *seconds* **Example:** `Device(config-profile)# tcp finwait-time 5`	(Optional) Specifies the length of time a TCP session will be managed after the Cisco firewall detects a finish (FIN)-exchange.
Step 13	**tcp idle-time** *seconds* **Example:** `Device(config-profile)# tcp idle-time 90`	(Optional) Configures the timeout for TCP sessions.

	Command or Action	Purpose
Step 14	**tcp max-incomplete host** *threshold* [**block-time** *minutes*] **Example:** `Device(config-profile)# tcp max-incomplete host 500 block-time 10`	(Optional) Specifies threshold and blocking time values for TCP host-specific Denial-of-Service (DoS) detection and prevention.
Step 15	**tcp synwait-time** *seconds* **Example:** `Device(config-profile)# tcp synwait-time 3`	(Optional) Specifies how long the software will wait for a TCP session to reach the established state before dropping the session.
Step 16	**tcp window-scale-enforcement loose** **Example:** `Device(config-profile)# tcp window-scale-enforcement loose`	(Optional) Disables the window scale option check in the parameter map for a TCP packet that has an invalid window scale option under the zone-based policy firewall.
Step 17	**udp idle-time** *seconds* **Example:** `Device(config-profile)# udp idle-time 75`	(Optional) Configures an idle timeout of UDP sessions that are going through the firewall.
Step 18	**end** **Example:** `Device(config-profile)# end`	Exits parameter map type inspect configuration mode and returns to privileged EXEC configuration mode.

Creating a URL Filter Parameter Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **parameter-map type urlfilter** *parameter-map-name*
4. **alert** {**on** | **off**}
5. **allow-mode** {**on** | **off**}
6. **audit-trail** {**on** | **off**}
7. **cache** *number*
8. **exclusive-domain** {**deny** | **permit**} *domain-name*
9. **max-request** *number-of-requests*
10. **max-resp-pak** *number-of-requests*
11. **server vendor** {**n2h2** | **websense**} {*ip-address* | *hostname* [**port** *port-number*]} [**outside**] [**log**] [**retrans** *retransmission-count*] [**timeout** *seconds*]
12. **source-interface** *interface-name*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**parameter-map type urlfilter** *parameter-map-name* **Example:** `Device(config)# parameter-map type urlfilter eng-network-profile`	Creates or modifies a parameter map for URL filtering parameters and enters parameter map type inspect configuration mode. **Note** This command is hidden depending on your release, but it continues to work. The **parameter-map type urlfpolicy** command can also be used to create URL filtering parameters for local, trend, Websense Internet filtering, and the N2H2 Internet blocking program. Depending on your release, use the URL filter policy rather than the URL filter action. All the use cases supported by the URL filter as an action are also supported by the URL filter policy. See the "Configuring a URL Filter Policy" section for more information.

	Command or Action	Purpose
Step 4	**alert** {**on** | **off**} **Example:** `Device(config-profile)# alert on`	(Optional) Enables Cisco stateful packet inspection alert messages that are displayed on the console.
Step 5	**allow-mode** {**on** | **off**} **Example:** `Device(config-profile)# allow-mode on`	(Optional) Enables the default mode of the filtering algorithm.
Step 6	**audit-trail** {**on** | **off**} **Example:** `Device(config-profile)# audit-trail on`	(Optional) Enables audit trail messages.
Step 7	**cache** *number* **Example:** `Device(config-profile)# cache 5`	(Optional) Controls how the URL filter handles the cache it maintains for HTTP servers.
Step 8	**exclusive-domain** {**deny** | **permit**} *domain-name* **Example:** `Device(config-profile)# exclusive-domain permit cisco.com`	(Optional) Adds a domain name to or from the exclusive domain list so that the Cisco firewall does not have to send lookup requests to the vendor server.
Step 9	**max-request** *number-of-requests* **Example:** `Device(config-profile)# max-request 80`	(Optional) Specifies the maximum number of outstanding requests that exist at a time.
Step 10	**max-resp-pak** *number-of-requests* **Example:** `Device(config-profile)# max-resp-pak 200`	(Optional) Specifies the maximum number of HTTP responses that the Cisco firewall can keep in its packet buffer.
Step 11	**server vendor** {**n2h2** | **websense**} {*ip-address* | *hostname* [**port** *port-number*]} [**outside**] [**log**] [**retrans** *retransmission-count*] [**timeout** *seconds*] **Example:** `Device(config-profile)# server vendor n2h2 10.193.64.22 port 3128 outside retrans 9 timeout 8`	Specifies the URL filtering server.
Step 12	**source-interface** *interface-name* **Example:** `Device(config-profile)# source-interface ethernet0`	(Optional) Specifies the interface whose IP address is used as the source IP address while making a TCP connection to the URL filter server (N2H2 or Websense).

	Command or Action	Purpose
Step 13	**end** **Example:** `Device(config-profile)# end`	Exits parameter map type inspect configuration mode and returns to privileged EXEC configuration mode.

Configuring a Layer 7 Protocol-Specific Parameter Map

Note Protocol-specific parameter maps are created only for instant messenger applications (AOL, ICQ, MSN Messenger, Yahoo Messenger, and Windows Messenger).

Before You Begin

To enable name resolution, you must enable the **ip domain name** command and the **ip name-server** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **parameter-map type protocol-info** *parameter-map-name*
4. **server** {**name** *string* [**snoop**] | **ip** {*ip-address* | **range** *ip-address-start ip-address-end*}}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**parameter-map type protocol-info** *parameter-map-name* **Example:** `Device(config)# parameter-map type protocol-info ymsgr`	Defines an application-specific parameter map and enters parameter map type inspect configuration mode.

	Command or Action	Purpose
Step 4	**server** {**name** *string* [**snoop**] | **ip** {*ip-address* | **range** *ip-address-start ip-address-end*}} **Example:** `Device(config-profile)# server name example1.example.com`	Configures a set of domain name system (DNS) servers with which a given instant messenger application will interact. **Note** If at least one server instance is not configured, the parameter map will not have any definitions to enforce; that is, the configured instant messenger policy cannot be enforced. **Note** To configure more than one set of servers, issue the **server** command multiple times within the parameter map of an instant messenger. Multiple entries are treated cumulatively.
Step 5	**end** **Example:** `Device(config-profile)# end`	Exits parameter map type inspect configuration mode and returns to privileged EXEC configuration mode.

Troubleshooting Tips

To display details of an Instant Messenger (IM) protocol-specific parameter map, use the **show parameter-map type protocol-info** command.

Configuring OoO Packet Processing Support in the Zone-Based Firewall Applications

Note When you configure a TCP-based Layer 7 policy for Deep Packet Inspection (DPI), Out-of-Order (OoO) packet processing is enabled by default. Use the **parameter-map type ooo global** command to configure the OoO packet support parameters or to disable OoO processing. Depending on your release, OoO processing was enabled for zone-based firewall and for Intrusion Prevention System (IPS)-shared sessions with Layer 4 match (**match protocol tcp**, **match protocol http**), and for any TCP-based Layer 7 packet ordering.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **parameter-map type ooo global**
4. **tcp reassembly alarm** {**on** | **off**}
5. **tcp reassembly memory limit** *memory-limit*
6. **tcp reassembly queue length** *queue-length*
7. **tcp reassembly timeout** *time-limit*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**parameter-map type ooo global** **Example:** `Device(config)# parameter-map type ooo global`	Configures OoO processing and enters parameter map type inspect configuration mode.
Step 4	**tcp reassembly alarm** {**on** | **off**} **Example:** `Device(config-profile)# tcp reassembly alarm on`	Specifies the alert message configuration.
Step 5	**tcp reassembly memory limit** *memory-limit* **Example:** `Device(config-profile)# tcp reassembly memory limit 2048`	Specifies the OoO box-wide buffer size.
Step 6	**tcp reassembly queue length** *queue-length* **Example:** `Device(config-profile)# tcp reassembly queue length 45`	Specifies the OoO queue length per TCP flow.
Step 7	**tcp reassembly timeout** *time-limit* **Example:** `Device(config-profile)# tcp reassembly timeout 34`	Specifies the OoO queue reassembly timeout value.
Step 8	**end** **Example:** `Device(config-profile)# end`	Exits parameter map type inspect configuration mode and returns to privileged EXEC configuration mode.

Configuring Intrazone Support in the Zone-Based Firewall Applications

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **zone-pair security** *zone-pair-name* [**source** *source-zone-name* **destination** *destination-zone-name*]
4. **exit**
5. **policy-map type inspect** *policy-map-name*
6. **class-map type inspect** *protocol-name* {**match-any** | **match-all**} *class-map-name*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**zone-pair security** *zone-pair-name* [**source** *source-zone-name* **destination** *destination-zone-name*] **Example:** `Device(config)# zone-pair security zonepair17 source zone8 destination zone8`	Specifies the name of the zone pair that is attached to an interface, the source zone for information passing, and the destination zone for information passing through this zone pair. • Enters security zone-pair configuration mode. **Note** To configure intrazone support, the source zone and the destination zone must be the same.
Step 4	**exit** **Example:** `Device(config-sec-zone-pair)# exit`	Exits security zone-pair configuration mode and returns to global configuration mode.
Step 5	**policy-map type inspect** *policy-map-name* **Example:** `Device(config)# policy-map type inspect my-pmap`	Specifies a policy map name and enters policy-map configuration mode.

	Command or Action	Purpose
Step 6	**class-map type inspect** *protocol-name* {**match-any** | **match-all**} *class-map-name* **Example:** `Device(config-pmap)# class-map type inspect aol match-any cmap1`	Specifies the firewall class map protocol and name.
Step 7	**end** **Example:** `Device(config-pmap)# end`	Exits policy map configuration mode and returns to privileged EXEC configuration mode.

Configuring Layer 7 Protocol-Specific Firewall Policies

Configure Layer 7 policy maps if you need extra provisioning for Layer 7 inspection modules. It is not necessary that you configure all Layer 7 policy maps specified in this section.

Perform one of the following tasks to configure a Layer 7, protocol-specific firewall policy:

Layer 7 Class Map and Policy Map Restrictions

- Deep packet inspection (DPI) class maps for Layer 7 can be used in inspect policy maps of the respective type. For example, **class-map type inspect http** can be used only in **policy-map type inspect http**.
- DPI policies require an **inspect** action at the parent level.
- A Layer 7 (DPI) policy map must be nested at the second level in a Layer 3 or Layer 4 inspect policy map, whereas a Layer 3 or Layer 4 inspect policy can be attached at the first level. Therefore, a Layer 7 policy map cannot be attached directly to a zone pair.
- If no action is specified in the hierarchical path of an inspect service policy, the packet is dropped. The traffic matching class-default in the top-level policy is dropped if there are no explicit actions configured in class-default. If the traffic does not match any class in a Layer 7 policy, the traffic is not dropped; control returns to the parent policy and subsequent actions (if any) in the parent policy are executed on the packet.
- Layer 7 policy maps include class maps only of the same type.
- You can specify the **reset** action only for TCP traffic; it resets the TCP connection.
- Depending on your release, removing a class that has a header with a regular expression from a Layer 7 policy map causes active HTTP sessions to reset. Prior to this change, when a class was removed from a Layer 7 policy map, the device is reloaded.

Configuring an HTTP Firewall Policy

To configure match criteria on the basis of an element within a parameter map, you must configure a parameter map as shown in the task "Creating an Inspect Parameter Map."

You must specify at least one match criterion; otherwise, the firewall policy will not be effective.

Configuring an HTTP Firewall Class Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map type inspect http** [**match-any** | **match-all**] *class-map-name*
4. **match response body java-applet**
5. **match req-resp protocol violation**
6. **match req-resp body length** {**lt** | **gt**} *bytes*
7. **match req-resp header content-type** {**violation** | **mismatch** | **unknown**}
8. **match** {**request** | **response** | **req-resp**} **header** [*header-name*] **count gt** *number*
9. **match** {**request** | **response** | **req-resp**} **header** [*header-name*] **length gt** *bytes*
10. **match request** {**uri** | **arg**} **length gt** *bytes*
11. **match request method** {**connect** | **copy** | **delete** | **edit** | **get** | **getattribute** | **getattributenames** | **getproperties**| **head** | **index** | **lock** | **mkdir** | **move** | **options** | **post** | **put** | **revadd** | **revlabel** | **revlog** | **revnum** | **save** | **setattribute** | **startrev** | **stoprev** | **trace** | **unedit** | **unlock**}
12. **match request port-misuse** {**im** | **p2p** | **tunneling** | **any**}
13. **match req-resp header transfer-encoding** {**chunked** | **compress** | **deflate** | **gzip** | **identity** | **all**}
14. **match** {**request** | **response** | **req-resp**} **header** [*header-name*] **regex** *parameter-map-name*
15. **match request uri regex** *parameter-map-name*
16. **match** {**request** | **response** | **req-resp**} **body regex** *parameter-map-name*
17. **match response status-line regex** *parameter-map-name*
18. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class-map type inspect http** [**match-any** | **match-all**] *class-map-name* **Example:** `Device(config)# class-map type inspect http http-class`	Creates a class map for the HTTP protocol so that you can enter match criteria and enters class-map configuration mode.
Step 4	**match response body java-applet** **Example:** `Device(config-cmap)# match response body java-applet`	(Optional) Identifies Java applets in an HTTP connection.
Step 5	**match req-resp protocol violation** **Example:** `Device(config-cmap)# match req-resp protocol violation`	(Optional) Configures an HTTP class map to allow HTTP messages to pass through the firewall or to reset the TCP connection when HTTP noncompliant traffic is detected.
Step 6	**match req-resp body length** {**lt** | **gt**} *bytes* **Example:** `Device(config-cmap)# match req-resp body length gt 35000`	(Optional) Configures an HTTP class map to use the minimum or maximum message size, in bytes, as a match criterion for permitting or denying HTTP traffic through the firewall.
Step 7	**match req-resp header content-type** {**violation** | **mismatch** | **unknown**} **Example:** `Device(config-cmap)# match req-resp header content-type mismatch`	(Optional) Configures an HTTP class map based on the content type of the HTTP traffic.
Step 8	**match** {**request** | **response** | **req-resp**} **header** [*header-name*] **count gt** *number* **Example:** `Device(config-cmap)# match req-resp header count gt 16`	(Optional) Configures an HTTP firewall policy to permit or deny HTTP traffic on the basis of both request and response messages whose header count does not exceed the specified maximum number of fields.
Step 9	**match** {**request** | **response** | **req-resp**} **header** [*header-name*] **length gt** *bytes* **Example:** `Device(config-cmap)# match response header length gt 50000`	(Optional) Permits or denies HTTP traffic based on the length of the HTTP request header.

	Command or Action	Purpose
Step 10	**match request** {**uri** | **arg**} **length gt** *bytes* **Example:** `Device(config-cmap)# match request uri length gt 500`	(Optional) Configures an HTTP firewall policy to use the Uniform Resource Identifier (URI) or argument length in the request message as a match criterion for permitting or denying HTTP traffic.
Step 11	**match request method** {**connect** | **copy** | **delete** | **edit** | **get** | **getattribute** | **getattributenames** | **getproperties**| **head** | **index** | **lock** | **mkdir** | **move** | **options** | **post** | **put** | **revadd** | **revlabel** | **revlog** | **revnum** | **save** | **setattribute** | **startrev** | **stoprev** | **trace** | **unedit** | **unlock**} **Example:** `Device(config-cmap)# match request method connect`	(Optional) Configures an HTTP firewall policy to use the request methods or the extension methods as a match criterion for permitting or denying HTTP traffic.
Step 12	**match request port-misuse** {**im** | **p2p** | **tunneling** | **any**} **Example:** `Device(config-cmap)# match request port-misuse any`	(Optional) Identifies applications misusing the HTTP port.
Step 13	**match req-resp header transfer-encoding** {**chunked** | **compress** | **deflate** | **gzip** | **identity** | **all**} **Example:** `Device(config-cmap)# match req-resp header transfer-encoding compress`	(Optional) Permits or denies HTTP traffic according to the specified transfer encoding of the message.
Step 14	**match** {**request** | **response** | **req-resp**} **header** [*header-name*] **regex** *parameter-map-name* **Example:** `Device(config-cmap)# match req-resp header regex non_ascii_regex`	(Optional) Configures HTTP firewall policy match criteria on the basis of headers that match the regular expression defined in a parameter map. • HTTP has two regular expression (regex) options. One combines the **header** keyword, **content-type** header name, and **regex** keyword and *parameter-map-name* argument. The other combines the **header** keyword, **regex** keyword, and *parameter-map-name* argument. • If the **header** and **regex** keywords are used with the *parameter-map-name* argument, the parameter map does not require a period and asterisk in front of the *parameter-map-name* argument. For example, either the "html" or ".*html" *parameter-map-name* argument can be configured. • If the **header** keyword is used with the content-type header name and **regex** keyword, then the parameter map name requires a period and asterisk (.*) in front of the

	Command or Action	Purpose
		parameter-map-name argument. For example, the *parameter-map-name* argument "html" is expressed as .*html. **Note** If the period and asterisk are added in front of "html" (.*html), the *parameter-map-name* argument works for both HTTP regex options. • The **mismatch** keyword is valid only for the **match response header content-type regex** command syntax for messages that need to be matched and that have a content-type header name mismatch. **Tip** It is a good practice to add ".*" to the **regex** *parameter-map-name* arguments that are not present at the beginning of a text string.
Step 15	**match request uri regex** *parameter-map-name* **Example:** `Device(config-cmap)# match request uri regex uri-regex-cm`	(Optional) Configures an HTTP firewall policy to permit or deny HTTP traffic on the basis of request messages whose URI or arguments (parameters) match a defined regular expression.
Step 16	**match** {**request** | **response** | **req-resp**} **body regex** *parameter-map-name* **Example:** `Device(config-cmap)# match response body regex body-regex`	(Optional) Configures a list of regular expressions that are to be matched against the body of the request, response, or both the request and response message.
Step 17	**match response status-line regex** *parameter-map-name* **Example:** `Device(config-cmap)# match response status-line regex status-line-regex`	(Optional) Specifies a list of regular expressions that are to be matched against the status line of a response message.
Step 18	**end** **Example:** `Device(config-cmap)# end`	(Optional) Exits class map configuration mode and returns to privileged EXEC mode.

Configuring an HTTP Firewall Policy Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map type inspect http** *policy-map-name*
4. **class-type inspect http** *http-class-name*
5. **allow**
6. **log**
7. **reset**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map type inspect http** *policy-map-name* **Example:** `Device(config)# policy-map type inspect http myhttp-policy`	Creates a Layer 7 HTTP policy map and enters policy-map configuration mode.
Step 4	**class-type inspect http** *http-class-name* **Example:** `Device(config-pmap)# class-type inspect http http-class`	Creates a class map for the HTTP protocol.
Step 5	**allow** **Example:** `Device(config-pmap)# allow`	(Optional) Allows a traffic class that matches the class.
Step 6	**log** **Example:** `Device(config-pmap)# log`	Generates log messages.

	Command or Action	Purpose
Step 7	**reset** **Example:** `Device(config-pmap)# reset`	(Optional) Resets a TCP connection if the data length of the Simple Mail Transfer Protocol (SMTP) body exceeds the value configured in the **class-map type inspect smtp** command.
Step 8	**end** **Example:** `Device(config-pmap)# end`	Exits policy-map configuration mode and returns to privileged EXEC mode.

Configuring a URL Filter Policy

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **parameter-map type urlfpolicy** {**local** | **n2h2** | **websense**} *parameter-map-name*
4. **exit**
5. **class-map type urlfilter** {*class-map-name* | **match-any** *class-map-name* | **n2h2** {*class-map-name* | **match-any** *class-map-name*} | **websense** {*class-map-name* | **match-any** *class-map-name*}}
6. **exit**
7. **policy-map type inspect urlfilter** *policy-map-name*
8. **service-policy urlfilter** *policy-map-name*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**parameter-map type urlfpolicy** {**local** | **n2h2** | **websense**} *parameter-map-name*	Configures the URL filter name related to the parameter map, which can include local, Websense, or N2H2

	Command or Action	Purpose
	Example: `Device(config)# parameter-map type urlfpolicy websense websense-param-map`	parameters and enters parameter map type inspect configuration mode.
Step 4	**exit** **Example:** `Device(config-profile)# exit`	Exits parameter map type inspect configuration mode and returns to global configuration mode.
Step 5	**class-map type urlfilter** {*class-map-name* | **match-any** *class-map-name* | **n2h2** {*class-map-name* | **match-any** *class-map-name*} | **websense** {*class-map-name* | **match-any** *class-map-name*}} **Example:** `Device(config)# class-map type urlfilter websense websense-param-map`	Configures the class map for the URL filter and enters class-map configuration mode.
Step 6	**exit** **Example:** `Device(config-cmap)# exit`	Exits class-map configuration mode and returns to global configuration mode.
Step 7	**policy-map type inspect urlfilter** *policy-map-name* **Example:** `Device(config)# policy-map type inspect urlfilter websense-policy`	Configures the URL filter policy and enters policy-map configuration mode.
Step 8	**service-policy urlfilter** *policy-map-name* **Example:** `Device(config-pmap)# service-policy urlfilter websense-policy`	Applies the URL filter policy under the inspect class as the service policy.
Step 9	**end** **Example:** `Device(config-pmap)# end`	Exits policy-map configuration mode and returns to privileged EXEC mode.

Configuring an IMAP Firewall Policy

Configuring an IMAP Class Map

Perform the following task to configure an Integrated Messaging Access Protocol (IMAP) class map:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip inspect name** *inspection-name protocol* [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**reset**] [**secure-login**] [**timeout** *seconds*]
4. **class-map type inspect imap** [**match-any**] *class-map-name*
5. **log**
6. **match invalid-command**
7. **match login clear-text**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip inspect name** *inspection-name protocol* [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**reset**] [**secure-login**] [**timeout** *seconds*] **Example:** `Device(config)# ip inspect name mail-guard imap`	Defines a set of inspection rules.
Step 4	**class-map type inspect imap** [**match-any**] *class-map-name* **Example:** `Device(config)# class-map type inspect imap imap-class`	Creates a class map for IMAP to enter the match criterion and enters class-map configuration mode.
Step 5	**log** **Example:** `Device(config-cmap)# log`	Generates log messages.
Step 6	**match invalid-command** **Example:** `Device(config-cmap)# match invalid-command`	(Optional) Locates invalid commands on an IMAP connection.

	Command or Action	Purpose
Step 7	**match login clear-text** **Example:** `Device(config-cmap)# match login clear-text`	(Optional) Locates nonsecure login when an IMAP server is used.
Step 8	**end** **Example:** `Device(config-cmap)# end`	Exits class-map configuration mode and returns to privileged EXEC configuration mode.

Configuring an IMAP Policy Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map type inspect imap** *policy-map-name*
4. **class-type inspect imap** *imap-class-name*
5. **log**
6. **reset**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map type inspect imap** *policy-map-name* **Example:** `Device(config)# policy-map type inspect imap myimap-policy`	Creates a Layer 3 Integrated Messaging Access Protocol (IMAP) policy map and enters policy-map configuration mode.

	Command or Action	Purpose
Step 4	**class-type inspect imap** *imap-class-name* **Example:** `Device(config-pmap)# class-type inspect imap pimap`	Creates a class map for the IMAP protocol.
Step 5	**log** **Example:** `Device(config-pmap)# log`	Generates log messages.
Step 6	**reset** **Example:** `Device(config-pmap)# reset`	(Optional) Resets a TCP connection if the data length of the Simple Mail Transfer Protocol (SMTP) body exceeds the value that you configured in the **class-map type inspect smtp** command.
Step 7	**end** **Example:** `Device(config-pmap)# end`	Exits policy-map configuration mode and returns to privileged EXEC mode.

Configuring an Instant Messenger Policy

Configuring an IM Class Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class map type inspect** {**aol** | **msnmsgr** | **ymsgr** | **icg** | **winmsgr**} [**match-any**] *class-map-name*
4. **match service** {**any** | **text-chat**}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class map type inspect** {**aol** | **msnmsgr** | **ymsgr** | **icg** | **winmsgr**} [**match-any**] *class-map-name* **Example:** `Device(config)# class map type inspect aol myaolclassmap`	Creates an Instant Messenger (IM) type class map so that you can begin adding match criteria and enters class-map configuration mode.
Step 4	**match service** {**any** | **text-chat**} **Example:** `Device(config-cmap)# match service text-chat`	(Optional) Creates a match criterion on the basis of text chat messages.
Step 5	**end** **Example:** `Device(config-cmap)# end`	Exits class-map configuration mode and returns to privileged EXEC mode.

Configuring an IM Policy Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy map type inspect** *protocol-name policy-map-name*
4. **class type inspect** {**aol** | **msnmsgr** | **ymsgr** | **icq** | **winmsgr**} *class-map-name*
5. **reset**
6. **log**
7. **allow**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy map type inspect** *protocol-name policy-map-name* **Example:** `Device(config)# policy map type inspect aol myaolpolicymap`	Creates an Instant Messenger (IM) policy map and enters policy-map configuration mode.
Step 4	**class type inspect** {**aol** | **msnmsgr** | **ymsgr** | **icq** | **winmsgr**} *class-map-name* **Example:** `Device(config-pmap)# class type inspect aol myaolclassmap`	Specifies a traffic class on which an action is to be performed. • *class-map-name*—This class map name should match the class map specified by using the **class-map type inspect** command.
Step 5	**reset** **Example:** `Device(config-pmap)# reset`	(Optional) Resets the connection.
Step 6	**log** **Example:** `Device(config-pmap)# log`	(Optional) Generates a log message for the matched parameters.
Step 7	**allow** **Example:** `Device(config-pmap)# allow`	(Optional) Allows the connection.
Step 8	**end** **Example:** `Device(config-pmap)# end`	Exits policy-map configuration mode and returns to privileged EXEC mode.

Configuring a Peer-to-Peer Policy

You can create a peer-to-peer (P2P) policy for the following P2P applications: eDonkey, FastTrack, Gnutella, and Kazaa Version 2.

Configuring a Peer-to-Peer Class Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class map type inspect** {**edonkey** | **fasttrack** | **gnutella** | **kazaa2**} [**match-any**] *class-map-name*
4. **match file-transfer** [*regular-expression*]
5. **match search-file-name** [*regular-expression*]
6. **match text-chat** [*regular-expression*]
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class map type inspect** {**edonkey** | **fasttrack** | **gnutella** | **kazaa2**} [**match-any**] *class-map-name* **Example:** `Device(config)# class map type inspect edonkey myclassmap`	Creates a peer-to-peer type class map so that you can begin adding match criteria and enters class-map configuration mode.
Step 4	**match file-transfer** [*regular-expression*] **Example:** `Device(config-cmap)# match file-transfer *`	(Optional) Matches file transfer connections within any supported peer-to-peer protocol. **Note** To specify that all file transfer connections should be identified by the traffic class, use "*" as the regular expression.
Step 5	**match search-file-name** [*regular-expression*] **Example:** `Device(config-cmap)# match search-file-name`	(Optional) Blocks filenames within a search request for clients using the eDonkey application. **Note** This command is applicable only for the eDonkey application.
Step 6	**match text-chat** [*regular-expression*] **Example:** `Device(config-cmap)# match text-chat`	(Optional) Blocks text chat messages between clients using the eDonkey peer-to-peer application. **Note** This command is applicable only for the eDonkey application.

	Command or Action	Purpose
Step 7	**end** **Example:** `Device(config-cmap)# end`	Exits class-map configuration mode and returns to privileged EXEC mode.

Configuring a Peer-to-Peer Policy Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy map type inspect p2p** *policy-map-name*
4. **class type inspect** {**edonkey** | **fasttrack** | **gnutella** | **kazaa2**} *class-map-name*
5. **reset**
6. **log**
7. **allow**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy map type inspect p2p** *policy-map-name* **Example:** `Device(config)# policy map type inspect p2p mypolicymap`	Creates a peer-to-peer policy map and enters policy-map configuration mode.
Step 4	**class type inspect** {**edonkey** | **fasttrack** | **gnutella** | **kazaa2**} *class-map-name* **Example:** `Device(config-pmap)# class type inspect edonkey myclassmap`	Specifies a traffic class on which an action is to be performed and enters policy-map configuration mode. • *class-map-name*—This class map name should match the class map specified in the **class-map type inspect** command.

	Command or Action	Purpose
Step 5	**reset** **Example:** `Device(config-pmap)# reset`	(Optional) Resets the connection.
Step 6	**log** **Example:** `Device(config-pmap)# log`	(Optional) Generates a log message for the matched parameters.
Step 7	**allow** **Example:** `Device(config-pmap)# allow`	(Optional) Allows the connection.
Step 8	**end** **Example:** `Device(config-pmap)# end`	Exits policy-map configuration mode and returns to privileged EXEC mode.

Configuring a POP3 Firewall Policy

Configuring a POP3 Firewall Class Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip inspect name** *inspection-name protocol* [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**reset**] [**secure-login**] [**timeout** *seconds*]
4. **class-map type inspect pop3** [**match-any**] *class-map-name*
5. **match invalid-command**
6. **match login clear-text**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip inspect name** *inspection-name protocol* [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**reset**] [**secure-login**] [**timeout** *seconds*] **Example:** `Device(config)# ip inspect name mail-guard pop3`	Defines a set of inspection rules.
Step 4	**class-map type inspect pop3** [**match-any**] *class-map-name* **Example:** `Device(config)# class-map type inspect pop3 pop3-class`	Creates a class map for the Post Office Protocol, Version 3 (POP3) protocol to enter match criteria and enters class-map configuration mode.
Step 5	**match invalid-command** **Example:** `Device(config-cmap)# match invalid-command`	(Optional) Locates invalid commands on a POP3 server.
Step 6	**match login clear-text** **Example:** `Device(config-cmap)# match login clear-text`	(Optional) Locates a nonsecure login when using a POP3 server.
Step 7	**end** **Example:** `Device(config-cmap)# end`	Exits class-map configuration mode and returns to privileged EXEC mode.

Configuring a POP3 Firewall Policy Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map type inspect pop3** *policy-map-name*
4. **class-type inspect pop3** *pop3-class-name*
5. **log**
6. **reset**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map type inspect pop3** *policy-map-name* **Example:** `Device(config)# policy-map type inspect pop3 mypop3-policy`	Creates a Layer 7 Post Office Protocol, Version 3 (POP3) policy map and enters policy-map configuration mode.
Step 4	**class-type inspect pop3** *pop3-class-name* **Example:** `Device(config-pmap)# class-type inspect pop3 pcl`	Creates a class map for the POP3 protocol.
Step 5	**log** **Example:** `Device(config-pmap)# log`	Generates log messages.
Step 6	**reset** **Example:** `Device(config-pmap)# reset`	(Optional) Resets a TCP connection if the data length of the Simple Mail Transfer Protocol (SMTP) body exceeds the value that you configured in the **class-map type inspect smtp** command.

	Command or Action	Purpose
Step 7	**end** **Example:** `Device(config-pmap)# end`	Exits policy-map configuration mode and returns to privileged EXEC mode.

Configuring an SMTP Firewall Policy

Configuring an SMTP Firewall Class Map

Note To enable inspection for extended SMTP (ESMTP) in a class map, use the **match protocol smtp extended** command. See the "Restrictions for Zone-Based Policy Firewall" section for more information on using this command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map type inspect smtp** [**match-all** | **match-any**] *class-map-name*
4. **match data-length gt** *max-data-value*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**class-map type inspect smtp** [**match-all** | **match-any**] *class-map-name* **Example:** `Device(config)# class-map type inspect smtp smtp-class`	Creates a class map for the Simple Mail Transfer Protocol (SMTP) protocol to enter match criteria and enters class-map configuration mode.
Step 4	**match data-length gt** *max-data-value* **Example:** `Device(config-cmap)# match data-length gt 200000`	Determines if the amount of data transferred in an SMTP connection is above the configured limit.
Step 5	**end** **Example:** `Device(config-cmap)# end`	Exits class-map configuration mode and returns to privileged EXEC mode.

Configuring an SMTP Firewall Policy Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map type inspect smtp** *policy-map-name*
4. **class-type inspect smtp** *smtp-class-name*
5. **reset**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**policy-map type inspect smtp** *policy-map-name* **Example:** `Device(config)# policy-map type inspect smtp mysymtp-policy`	Creates a Layer 7 Simple Mail Transfer Protocol (SMTP) policy map and enters policy-map configuration mode.
Step 4	**class-type inspect smtp** *smtp-class-name* **Example:** `Device(config-pmap)# class-type inspect smtp sc`	Configures inspection parameters for an SMTP protocol.
Step 5	**reset** **Example:** `Device(config-pmap)# reset`	(Optional) Resets the TCP connection if the data length of the SMTP body exceeds the value that you configured in the **class-map type inspect smtp** command.
Step 6	**end** **Example:** `Device(config-pmap)# end`	Exits policy-map configuration mode and returns to privileged EXEC mode.

Configuring a SUNRPC Firewall Policy

Note If you are inspecting a remote-procedure call (RPC) protocol (that is, you have specified the **match protocol sunrpc** command in the Layer 4 class map), the Layer 7 SUNRPC policy map is required.

Configuring a SUNRPC Firewall Class Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map type inspect sunrpc** [**match-any**] *class-map-name*
4. **match program-number** *program-number*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class-map type inspect sunrpc** [**match-any**] *class-map-name* **Example:** `Device(config)# class-map type inspect sunrpc long-urls`	Creates a class map for the SUNRPC protocol to enter match criteria and enters class-map configuration mode.
Step 4	**match program-number** *program-number* **Example:** `Device(config-cmap)# match program-number 2345`	(Optional) Specifies the allowed remote-procedure call (RPC) protocol program number as a match criterion.
Step 5	**end** **Example:** `Device(config-cmap)# end`	Exits policy-map configuration mode and returns to privileged EXEC mode.

Configuring a SUNRPC Firewall Policy Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map type inspect sunrpc** *policy-map-name*
4. **class-type inspect sunrpc** *sunrpc-class-name*
5. **allow** [**wait-time** *minutes*]
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map type inspect sunrpc** *policy-map-name* **Example:** `Device(config)# policy-map type inspect sunrpc my-rpc-policy`	Creates a Layer 7 SUNRPC policy map and enters policy-map configuration mode.
Step 4	**class-type inspect sunrpc** *sunrpc-class-name* **Example:** `Device(config-pmap)# class-type inspect sunrpc cs1`	Configures inspection parameters for the SUNRPC protocol.
Step 5	**allow** [**wait-time** *minutes*] **Example:** `Device(config-pmap)# allow wait-time 10`	(Optional) Allows the configured program number. • Specifies the wait time in minutes to keep a keyhole open in the firewall to allow subsequent connections from the same source address to the same destination address and port. The default wait time is zero minutes. This keyword is available only for the remote-procedure call (RPC) protocol.
Step 6	**end** **Example:** `Device(config-pmap)# end`	Exits policy-map configuration mode and returns to privileged EXEC mode.

Configuring an MSRPC Firewall Policy

Note If you are inspecting an remote-procedure call (RPC) protocol (that is, you have specified the **match protocol msrpc** command in the Layer 4 class map), the Layer 7 Microsoft Remote Procedure Call (MSRPC) policy map is required.

Perform the following task to configure an MSRPC firewall policy:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **parameter-map type protocol-info msrpc** *parameter-map-name*
4. **timeout** *seconds*
5. **exit**
6. **class-map type inspect match-any** *class-map-name*
7. **match protocol msrpc**
8. **match protocol msrpc-smb-netbios**
9. **exit**
10. **policy-map type inspect** *policy-map-name*
11. **class type inspect** *class-map-name*
12. **inspect**
13. **exit**
14. **class class-default**
15. **drop**
16. **exit**
17. **exit**
18. **zone security** *security-zone-name*
19. **exit**
20. **zone security** *security-zone-name*
21. **exit**
22. **zone-pair security** *zone-pair-name* **source** *source-zone* **destination** *destination-zone*
23. **service-policy type inspect** *policy-map-name*
24. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**parameter-map type protocol-info msrpc** *parameter-map-name* **Example:** `Device(config)# parameter-map type protocol-info msrpc para-map`	Defines an application-specific parameter map and enters parameter map type inspect configuration mode.
Step 4	**timeout** *seconds* **Example:** `Device(config-profile)# timeout 60`	Configures the MSRPC endpoint mapper (EPM) timeout.
Step 5	**exit** **Example:** `Device(config-profile)# exit`	Exits parameter map type inspect configuration mode and returns to global configuration mode.
Step 6	**class-map type inspect match-any** *class-map-name* **Example:** `Device(config)# class-map type inspect match-any c-map`	Creates an inspect type class map for the traffic class and enters class-map configuration mode.
Step 7	**match protocol msrpc** **Example:** `Device(config-cmap)# match protocol msrpc`	Configures match criteria for a class map on the basis of a specified protocol. • Only Cisco stateful packet inspection-supported protocols can be used as match criteria in inspect type class maps.
Step 8	**match protocol msrpc-smb-netbios** **Example:** `Device(config-cmap)# match protocol msrpc-smb-netbios`	Configures match criteria for a class map on the basis of a specified protocol. • Only Cisco stateful packet inspection-supported protocols can be used as match criteria in inspect type class maps.
Step 9	**exit** **Example:** `Device(config-cmap)# exit`	Exits class-map configuration mode and returns to global configuration mode.
Step 10	**policy-map type inspect** *policy-map-name* **Example:** `Device(config)# policy-map type inspect p-map`	Creates a Layer 3 and Layer 4 inspect type policy map and enters policy-map configuration mode.
Step 11	**class type inspect** *class-map-name* **Example:** `Device(config-pmap)# class type inspect c-map`	Specifies the traffic (class) on which an action is to be performed and enters policy-map class configuration mode.

	Command or Action	Purpose
Step 12	**inspect** **Example:** `Device(config-pmap-c)# inspect`	Enables Cisco stateful packet inspection.
Step 13	**exit** **Example:** `Device(config-pmap-c)# exit`	Exits policy-map class configuration mode and returns to policy-map configuration mode.
Step 14	**class class-default** **Example:** `Device(config-pmap)# class class-default`	Specifies the matching of the system default class and enters policy-map class configuration mode. • If the system default class is not specified, unclassified packets are matched.
Step 15	**drop** **Example:** `Device(config-pmap-c)# drop`	Drops packets that match a defined class.
Step 16	**exit** **Example:** `Device(config-pmap-c)# exit`	Exits policy-map class configuration mode and returns to policy-map configuration mode.
Step 17	**exit** **Example:** `Device(config-pmap)# exit`	Exits policy-map configuration mode and returns to global configuration mode.
Step 18	**zone security** *security-zone-name* **Example:** `Device(config)# zone security in-zone`	Creates a security zone to which interfaces can be assigned and enters security zone configuration mode.
Step 19	**exit** **Example:** `Device(config-sec-zone)# exit`	Exits security zone configuration mode and returns to global configuration mode.
Step 20	**zone security** *security-zone-name* **Example:** `Device(config)# zone security out-zone`	Creates a security zone to which interfaces can be assigned and enters security zone configuration mode.
Step 21	**exit** **Example:** `Device(config-sec-zone)# exit`	Exits security zone configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 22	**zone-pair security** *zone-pair-name* **source** *source-zone* **destination** *destination-zone* **Example:** `Device(config)# zone-pair security in-out source in-zone destination out-zone`	Creates a zone pair and enters security zone-pair configuration mode. **Note** To apply a policy, you must configure a zone pair.
Step 23	**service-policy type inspect** *policy-map-name* **Example:** `Device(config-sec-zone-pair)# service-policy type inspect p-map`	Attaches a firewall policy map to the destination zone pair. **Note** If a policy is not configured between a pair of zones, traffic is dropped by default.
Step 24	**end** **Example:** `Device(config-sec-zone-pair)# end`	Exits security zone-pair configuration mode and returns to privileged EXEC mode.

Creating Security Zones and Zone Pairs and Attaching a Policy Map to a Zone Pair

You need two security zones to create a zone pair. However, you can create only one security zone and use a system-defined security zone called “self.” Note that if you select a self zone, you cannot configure inspect policing.

Use this process to complete the following tasks:

- Assign interfaces to security zones.
- Attach a policy map to a zone pair.
- Create at least one security zone.
- Define zone pairs.

Tip Before you create zones, think about what should constitute the zones. The general guideline is that you should group interfaces that are similar when they are viewed from a security perspective.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **zone security** *zone-name*
4. **description** *line-of-description*
5. **exit**
6. **zone-pair security** *zone-pair name* [**source** *source-zone-name* | **self**] **destination** [**self** | *destination-zone-name*]
7. **description** *line-of-description*
8. **exit**
9. **interface** *type number*
10. **zone-member security** *zone-name*
11. **exit**
12. **zone-pair security** *zone-pair-name* [**source** *source-zone-name* | **self**] **destination** [**self** | *destination-zone-name*]
13. **service-policy type inspect** *policy-map-name*
14. **platform inspect match-statistics per-filter**
15. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**zone security** *zone-name* **Example:** `Device(config)# zone security z1`	Creates a security zone to which interfaces can be assigned and enters security zone configuration mode.
Step 4	**description** *line-of-description* **Example:** `Device(config-sec-zone)# description Internet Traffic`	(Optional) Describes the zone.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device(config-sec-zone)# exit`	Exits security zone configuration mode and returns to global configuration mode.
Step 6	**zone-pair security** *zone-pair name* [**source** *source-zone-name* | **self**] **destination** [**self** | *destination-zone-name*] **Example:** `Device(config)# zone-pair security zp source z1 destination z2`	Creates a zone pair and enters security zone-pair configuration mode. **Note** To apply a policy, you must configure a zone pair.
Step 7	**description** *line-of-description* **Example:** `Device(config-sec-zone-pair)# description accounting network`	(Optional) Describes the zone pair.
Step 8	**exit** **Example:** `Device(config-sec-zone-pair)# exit`	Exits security zone-pair configuration mode and returns to global configuration mode.
Step 9	**interface** *type number* **Example:** `Device(config)# interface ethernet 0`	Configures an interface and enters interface configuration mode.
Step 10	**zone-member security** *zone-name* **Example:** `Device(config-if)# zone-member security zone1`	Assigns an interface to a specified security zone. **Note** When you make an interface a member of a security zone, all traffic in and out of that interface (except traffic bound for the device or initiated by the device) is dropped by default. To let traffic through the interface, you must make the zone part of a zone pair to which you should apply a policy. If the policy permits traffic, traffic can flow through that interface.
Step 11	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 12	**zone-pair security** *zone-pair-name* [**source** *source-zone-name* | **self**] **destination** [**self** | *destination-zone-name*] **Example:** `Device(config)# zone-pair security zp source z1 destination z2`	Creates a zone pair and enters security zone-pair configuration mode.

	Command or Action	Purpose
Step 13	**service-policy type inspect** *policy-map-name* **Example:** `Device(config-sec-zone-pair)# service-policy type inspect p2`	Attaches a firewall policy map to the destination zone pair. **Note** If a policy is not configured between a pair of zones, traffic is dropped by default.
Step 14	**platform inspect match-statistics per-filter** **Example:** `Device(config-sec-zone-pair)# platform inspect match-statistics per-filter`	Enables zone-based firewall per-filter statistics. **Note** To enable per-filter statistics on the device, do the following: • RELOAD the device. • OR Remove all the service-policies and re-apply the changes to the statistics. To activate the **platform inspect match-statistics per-filter** command, re-apply all service-policies.
Step 15	**end** **Example:** `Device(config-sec-zone-pair)# end`	Exits security zone-pair configuration mode and returns to privileged EXEC mode.

Configuration Examples for Zone-Based Policy Firewalls

Example: Configuring Layer 3 and Layer 4 Firewall Policies

The following example shows a Layer 3 or Layer 4 top-level policy. The traffic is matched to the access control list (ACL) 199 and deep-packet HTTP inspection is configured. Configuring the **match access-group 101** enables Layer 4 inspection. As a result, Layer 7 inspection is omitted unless the class-map is of type **match-all**.

```
class-map type inspect match-all http-traffic
 match protocol http
 match access-group 101
!
policy-map type inspect mypolicy
 class type inspect http-traffic
  inspect
  service-policy http http-policy
```

Example: Adding WAN to self-zone and self-zone to WAN

The following example shows that a policy is not required to pass all Layer 2 Tunneling Protocol (L2TP) traffic to a router as the traffic allowed is destined to the router or the traffic is originated from the router.

However, in case we do not want all traffic to pass on to the router, and a policy is required to be configured for self-zone, we add WAN to the self-zone and self-zone to WAN to allow the L2TP traffic.

To allow the L2TP traffic, we need to use the below ACL in the classmap for the L2TP traffic:

```
ip access-list extended wan-self-pass
 permit udp any host 192.168.255.254 eq 1701

ip access-list extended self-wan-pass
 permit udp host 192.168.255.254 eq 1701 any
```

Example: Configuring Layer 7 Protocol-Specific Firewall Policies

The following example shows how to match HTTP sessions that have a URL length greater than 500. The Layer 7 policy action **reset** is configured.

```
class-map type inspect http long-urls
 match request uri length gt 500
policy-map type inspect http http-policy
 class type inspect http long-urls
  reset
```

The following example shows how to enable inspection for Extended SMTP (ESMTP) by including the **extended** keyword:

```
class-map type inspect c1
 match protocol smtp extended
policy-map type inspect p1
 class type inspect c1
  inspect
```

The **service-policy type inspect smtp** command is optional and can be entered after the **inspect** command.

Example: Configuring a URL Filter Policy

```
parameter-map type urlfpolicy websense-param-map
class-map type urlfilter websense websense-param-map
policy-map type inspect urlfilter websense-policy
service-policy urlfilter websense-policy
```

Example: Configuring a URL Filter Policy for Websense

Example: Websense Server Configuration

```
parameter-map type urlfpolicy websense websense-param-map
 server fw21-ss1-bldr.example.com timeout 30
 source-interface Loopback0
 truncate script-parameters
 cache-size maximum-entries 100
 cache-entry-lifetime 1
 block-page redirect-url http://abc.example.com
```

Example: Configuring the Websense Class Map

```
class-map type urlfilter websense match-any websense-class
 match server-response any
```

Example: Configuring the Websense URL Filter Policy

```
policy-map type inspect urlfilter websense-policy
 parameter type urlfpolicy websense websense-param-map
 class type urlfilter websense websense-class
  server-specified-action
  log
```

Example: Creating Security Zones and Zone Pairs and Attaching a Policy Map to a Zone Pair

Example: Creating a Security Zone

The following example shows how to create security zone z1, which is called finance department networks, and security zone z2, which is called engineering services network:

```
zone security z1
 description finance department networks
!
zone security z2
 description engineering services network
```

Example: Creating Zone Pairs

The following example shows how to create zones z1 and z2 and specifies that the firewall policy map is applied in zone z2 for traffic flowing between zones:

```
zone-pair security zp source z1 destination z2
service-policy type inspect p1
```

Example: Assigning an Interface to a Security Zone

The following example shows how to attach Ethernet interface 0 to zone z1 and Ethernet interface 1 to zone z2:

```
interface ethernet0
 zone-member security z1
!
interface ethernet1
 zone-member security z2
```

Example: Protocol Match Data Not Incrementing for a Class Map

The following configuration example causes the match counter problem in the **show policy-map type inspect zone-pair** command output:

```
class-map type inspect match-any y
 match protocol tcp
 match protocol icmp
```

```
class-map type inspect match-all x
 match class y
```

However, cumulative counters for the configuration are displayed in the **show policy-map type inspect zone-pair** command output if the class map matches any class map:

```
Device# show policy-map type inspect zone session

policy exists on zp zp
 Zone-pair: zp
  Service-policy inspect : fw
    Class-map: x (match-any)
      Match: class-map match-any y
        2 packets, 48 bytes   <======= Cumulative class map counters are incrementing.
        30 second rate 0 bps
        Match: protocol tcp
          0 packets, 0 bytes     <==== The match for the protocol is not incrementing.
          30 second rate 0 bps
        Match: protocol icmp
          0 packets, 0 bytes
          30 second rate 0 bps
   Inspect
      Number of Established Sessions = 1
      Established Sessions
        Session 53105C0 (10.1.1.2:19180)=>(172.16.1.2:23) telnet:tcp SIS_OPEN
          Created 00:00:02, Last heard 00:00:02
          Bytes sent (initiator:responder) [30:69]
    Class-map: class-default (match-any)
      Match: any
      Drop
        0 packets, 0 bytes
```

Example: Zone-Based Firewall Per-filter Statistics

The following configuration example shows how to prevent memory shortage when a large number of firewall filters are created. To prevent memory shortage, you can enable the zone-based firewall per-filter statistics with the **platform inspect match-statistics per-filter** command. In the example, for each filter (ACL or UDP), there are statistics available for the number of packets and the number of bytes traversed through zone-based firewall.

```
Device# show policy-map type inspect zone-pair ogacl_zp
 Zone-pair: ogacl_zp
  Service-policy inspect : ogacl_pm
Class-map: ogacl_cm (match-any)
  Match: access-group name ogacl
        xxx packets, xxx bytes
      Match: protocol udp
        xxx packets, xxx bytes
```

Note Per-filter statistics are available only for match-any filters and are not applicable for match-all cases.

Note For Cisco IOS XE 16.3 and Cisco IOS XE 16.4 releases, to enable per-filter statistics, either reload the device or remove the service-policies and then reapply the service policies on the zone pair before the **platform inspect match-statistics per-filter** command is activated.

For Cisco IOS XE 3.17 release, you must save the configuration and reload the system to activate this command.

Note Similarly, to disable per-filter statistics, either reload the device or remove the service-policies and then reapply the service policies on the zone pair.

To check the TCAM memory used in a device, use the **show platform hardware qfp active classification feature-manager shm-stats-counter** command.

```
Device# show platform hardware qfp active classification feature-manager shm-stats-counter
    Shared Memory Information:
    Total shared memory size: 16777216
    Used shared memory size: 14703656
```

Note If traffic drops or per-filter statistics counters are not displayed, then probabilty is the TCAM shared memory used is more than 75% of the total TCAM.

Note If the shared memory used in the device is more than 75% of the capacity, the following warning message is displayed :

```
%CPP_FM-3-CPP_FM_TCAM_WARNING: SIP1: cpp_sp_svr: TCAM limit exceeded: Already used 75
percent shared memory for per-filter stats.
```

If the shared memory used in the device is 100%, the following warning message is displayed:

```
%CPP_FM-3-CPP_FM_TCAM_WARNING: SIP1: cpp_sp_svr: TCAM limit exceeded: Shared memory for
per-filter stats overflow!
```

Additional References for Zone-Based Policy Firewalls

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Security commands	• Cisco IOS Security Command Reference: Commands A to C • Cisco IOS Security Command Reference: Commands D to L • Cisco IOS Security Command Reference: Commands M to R • Cisco IOS Security Command Reference: Commands S to Z
Quality of service commands	Cisco IOS Quality of Service Solutions Command Reference

Standards and RFCs

Standard & RFC	Title
RFC 1950	ZLIB Compressed Data Format Specification version 3.3
RFC 1951	DEFLATE Compressed Data Format Specification version 1.3
RFC 2616	Hypertext Transfer Protocol—HTTP/1.1

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Zone-Based Policy Firewalls

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1: Feature Information for Zone-Based Policy Firewalls

Feature Name	Releases	Feature Information
Application Inspection and Control for HTTP—Phase 2	12.4(9)T	The Application Inspection and Control for HTTP—Phase 2 feature extends support for HTTP application firewall policies. The following commands were introduced or modified by this feature: **regexmatch body regex**, **match header count**, **match header length**, **match header regex**, **match request length**, **match request**, **match response status-line regex**.

Feature Name	Releases	Feature Information
E-mail Inspection Engine	15.1(1)S	The E-mail Inspection Engine feature allows users to inspect POP3, IMAP, and E/SMTP e-mail traffic contained in SSL VPN tunneled connections that traverse the Cisco device.
P2P Application Inspection and Control—Phase 1	12.4(9)T 12.4(20)T 15.3(1)T	The P2P Application Inspection and Control—Phase 1 feature introduces support for identifying and enforcing a configured policy for the following peer-to-peer applications: eDonkey, FastTrack, Gnutella Version 2, and Kazaa Version 2. Support for identifying and enforcing a configured policy for the following Instant Messenger (IM) applications is also introduced: AOL, MSN Messenger, and Yahoo Messenger. In Release 12.4(20)T, support was added for the following applications: H.323, VoIP, and SIP. In Release 12.4(20)T, support for the following IM applications was also added: ICQ and Windows Messenger. The following commands were introduced or modified by this feature: **class-map type inspect**, **class type inspect**, **clear parameter-map type protocol-info**, **debug policy-firewall**, **match file-transfer**, **match protocol (zone)**, **match search-file-name**, **match service**, **match text-chat**, **parameter-map type**, **policy-map type inspect**, **server (parameter-map)**, **show parameter-map type protocol-info**. In 15.3(1)T and later releases, the following peer-to-peer protocols are deprecated: • BitTorrent • DirectConnect • eDonkey • FastTrack • Gnutella Version 2 • Kazaa Version 2 • WinMX

Feature Name	Releases	Feature Information
Rate-Limiting Inspected Traffic	12.4(9)T	The Rate-Limiting Inspected Traffic feature allows users to rate limit traffic within a Cisco firewall (inspect) policy. Also, users can limit the absolute number of sessions that can exist on a zone pair. The following commands were introduced by this feature: **police (zone policy)** and **sessions maximum**.
Zone-Based Policy Firewalls	12.4(6)T	The Zone-Based Policy Firewall feature provides a Cisco unidirectional firewall policy between groups of interfaces known as zones. The following commands were introduced or modified by this feature: **class-map type inspect**, **class type inspect**, **clear parameter-map type protocol-info**, **debug policy-firewall**, **match body regex**, **match file-transfer**, **match header count**, **match header length**, **match header regex**, **match protocol** (**zone**), **match request length**, **match request regex**, **match response status-line regex**, **match search-file-name**, **match service**, **match text-chat**, **parameter-map type**, **policy-map type inspect**, **server (parameter-map)**, **service-policy (policy-map)**, **service-policy type inspect**, **show parameter-map type protocol-info**.
Zone-Based Firewall—Default Zone	15.6(1)T	The Zone-Based Firewall— Default Zone feature introduces a default zone that enables a firewall policy to be configured on a zone pair that consist of a zone and a default zone. Any interface without explicit zone membership belongs to a default zone. The following commands were introduced by this feature: **zone pair security**, **zone security default**.
Zone-Based Firewall Support for Microsoft Remote Procedure Call (MSRPC)	15.1(4)M	The Zone-Based Firewall Support for MSRPC feature introduces zone-based policy firewall support for MSRPC.
Zone-Based Firewall Support of Multipoint TCP	15.4(3)M	Multipoint TCP seamlessly works with zone-based firewall Layer 4 inspection. Multipoint TCP does not work with application layer gateways (ALGs) and application inspection and control (AIC).

Feature Name	Releases	Feature Information
Zone-Based Firewall Usability and Manageability	15.0(1)M 15.1(1)T	The Zone-Based Firewall Usability and Manageability features covered in this document are out-of-order (OoO) packet processing support in zone-based firewalls, intrazone support in zone-based firewalls, and enhanced debug capabilities. The following commands were introduced or modified by this feature: **clear ip ips statistics**, **debug cce dp named-db inspect**, **debug policy-firewall**, **debug ip virtual-reassembly list**, **parameter-map type ooo global**, **show parameter-map type ooo global**, **zone-pair security**. Depending on your release, the following commands were introduced or modified: **class-map type inspect**, **clear policy-firewall**, **log (parameter-map type)**, **match request regex**, **parameter-map type inspect**, **show parameter-map type inspect**, **show policy-firewall config**, **show policy-firewall mib**, **show policy-firewall sessions**, **show policy-firewall stats**, **show policy-firewall summary-log**.