Connection Settings

This chapter describes how to configure connection settings for connections that go through the ASA, or for management connections that go to the ASA.

- What Are Connection Settings?, on page 1
- Configure Connection Settings, on page 2
- Monitoring Connections, on page 14
- History for Connection Settings, on page 15

What Are Connection Settings?

Connection settings comprise a variety of features related to managing traffic connections, such as a TCP flow through the ASA. Some features are named components that you would configure to supply specific services.

Connection settings include the following:

- **Global timeouts for various protocols**—All global timeouts have default values, so you need to change them only if you are experiencing premature connection loss.
- **Connection timeouts per traffic class**—You can override the global timeouts for specific types of traffic using service policies. All traffic class timeouts have default values, so you do not have to set them.
- **Connection limits and TCP Intercept**—By default, there are no limits on how many connections can go through (or to) the ASA. You can set limits on particular traffic classes using service policy rules to protect servers from denial of service (DoS) attacks. Particularly, you can set limits on embryonic connections (those that have not finished the TCP handshake), which protects against SYN flooding attacks. When embryonic limits are exceeded, the TCP Intercept component gets involved to proxy connections and ensure that attacks are throttled.
- **Dead Connection Detection (DCD)**—If you have persistent connections that are valid but often idle, so that they get closed because they exceed idle timeout settings, you can enable Dead Connection Detection to identify idle but valid connections and keep them alive (by resetting their idle timers). Whenever idle times are exceeded, DCD probes both sides of the connection to see if both sides agree the connection is valid. The **show service-policy** command output includes counters to show the amount of activity from DCD.
- **TCP sequence randomization**—Each TCP connection has two initial sequence numbers (ISN): one generated by the client and one generated by the server. By default, the ASA randomizes the ISN of the TCP SYN passing in both the inbound and outbound directions. Randomization prevents an attacker

from predicting the next ISN for a new connection and potentially hijacking the new session. You can disable randomization per traffic class if desired.

- **TCP Normalization**—The TCP Normalizer protects against abnormal packets. You can configure how some types of packet abnormalities are handled by traffic class.
- **TCP State Bypass**—You can bypass TCP state checking if you use asymmetrical routing in your network.

Configure Connection Settings

Connection limits, timeouts, TCP Normalization, TCP sequence randomization, and decrementing time-to-live (TTL) have default values that are appropriate for most networks. You need to configure these connection settings only if you have unusual requirements, your network has specific types of configuration, or if you are experiencing unusual connection loss due to premature idle timeouts.

Other connection-related features are not enabled. You would configure these services on specific traffic classes only, and not as a general service. These features include the following: TCP Intercept, TCP State Bypass, Dead Connection Detection (DCD).

The following general procedure covers the gamut of possible connection setting configurations. Pick and choose which to implement based on your needs.

Procedure

Step 1 Configure Global Timeouts, on page 2. These settings change the default idle timeouts for various protocols for all traffic that passes through the device. If you are having problems with connections being reset due to premature timeouts, first try changing the global timeouts.

Step 2 Protect Servers from a SYN Flood DoS Attack (TCP Intercept), on page 5. Use this procedure to configure TCP Intercept.

Step 3 Customize Abnormal TCP Packet Handling (TCP Maps, TCP Normalizer), on page 6, if you want to alter the default TCP Normalization behavior for specific traffic classes.

Step 4 Bypass TCP State Checks for Asynchronous Routing (TCP State Bypass), on page 8, if you have this type of routing environment.

Step 5 Disable TCP Sequence Randomization, on page 11, if the default randomization is scrambling data for certain connections.

Step 6 Configure Connection Settings for Specific Traffic Classes (All Services), on page 12. This is a catch-all procedure for connection settings. These settings can override the global defaults for specific traffic classes using service policy rules. You also use these rules to customize TCP Normalizer, change TCP sequence randomization, decrement time-to-live on packets, and implement other optional features.

Configure Global Timeouts

You can set the global idle timeout durations for the connection and translation slots of various protocols. If the slot has not been used for the idle time specified, the resource is returned to the free pool.

Changing the global timeout sets a new default timeout, which in some cases can be overridden for particular traffic flows through service policies.

Procedure

Step 1 Choose **Configuration** > **Firewall** > **Advanced** > **Global Timeouts**.

Step 2 Configure the timeouts by checking the boxes for timeouts you want to change and entering the new value.

All durations are displayed in the format *hh*:*mm*:*ss*, with a maximum duration of 1193:0:0 in most cases. In all cases, except for Authentication absolute and Authentication inactivity, unchecking the check boxes returns the timeout to the default value. For those two cases, clearing the check box means to reauthenticate on every new connection.

Enter 0 to disable a timeout.

- **Connection**—The idle time until a connection slot is freed. This duration must be at least 5 minutes. The default is 1 hour.
- **Half-closed**—The idle time until a TCP half-closed connection closes. A connection is considered half-closed if both the FIN and FIN-ACK have been seen. If only the FIN has been seen, the regular connection timeout applies. The minimum is 30 seconds. The default is 10 minutes.
- **UDP**—The idle time until a UDP connection closes. This duration must be at least 1 minute. The default is 2 minutes.
- **ICMP**—The idle time after which general ICMP states are closed. The default (and minimum) is 2 seconds.
- **H.323**—The idle time after which H.245 (TCP) and H.323 (UDP) media connections close. The default (and minimum) is 5 minutes. Because the same connection flag is set on both H.245 and H.323 media connections, the H.245 (TCP) connection shares the idle timeout with the H.323 (RTP and RTCP) media connection.
- **H.225**—The idle time until an H.225 signaling connection closes. The default is 1 hour. To close a connection immediately after all calls are cleared, a timeout of 1 second (0:0:1) is recommended.
- **MGCP**—The idle time after which an MGCP media connection is removed. The default is 5 minutes, but you can set it as low as 1 second.
- **MGCP PAT**—The idle time after which an MGCP PAT translation is removed. The default is 5 minutes. The minimum time is 30 seconds.
- **TCP Proxy Reassembly**—The idle timeout after which buffered packets waiting for reassembly are dropped, between 0:0:10 and 1193:0:0. The default is 1 minute (0:1:0).
- **Floating Connection**—When multiple routes exist to a network with different metrics, the system uses the one with the best metric at the time of connection creation. If a better route becomes available, then this timeout lets connections be closed so a connection can be reestablished to use the better route. The default is 0 (the connection never times out). To make it possible to use better routes, set the timeout to a value between 0:0:30 and 1193:0:0.
- **SUNRPC**—The idle time until a SunRPC slot is freed. This duration must be at least 1 minute. The default is 10 minutes.
- **SIP**—The idle time until a SIP signaling port connection closes. This duration must be at least 5 minutes. The default is 30 minutes.

- **SIP Media**—The idle time until a SIP media port connection closes. This duration must be at least 1 minute. The default is 2 minutes. The SIP media timer is used for SIP RTP/RTCP with SIP UDP media packets, instead of the UDP inactivity timeout.
- **SIP Provisional Media**—The timeout value for SIP provisional media connections, between 1 and 30 minutes. The default is 2 minutes.
- **SIP Invite**—The idle time after which pinholes for PROVISIONAL responses and media xlates will be closed, between 0:1:0 and 00:30:0. The default is 3 minutes (0:3:0).
- **SIP Disconnect**—The idle time after which SIP session is deleted if the 200 OK is not received for a CANCEL or a BYE message, between 0:0:1 and 0:10:0. The default is 2 minutes (0:2:0).
- **Authentication absolute**—The duration until the authentication cache times out and users have to reauthenticate a new connection. This timer is used in cut-through proxy only, which is a AAA rule. This duration must be shorter than the Translation Slot timeout. The system waits until the user starts a new connection to prompt again. Before you disable caching to force authentication on every new connection, consider the following limitations.
  - Do not set this value to 0 if passive FTP is used on the connections.
  - When Authentication Absolute is 0, HTTPS authentication may not work. If a browser initiates multiple TCP connections to load a web page after HTTPS authentication, the first connection is permitted through, but subsequent connections trigger authentication. As a result, users are continuously presented with an authentication page, even after successful authentication. To work around this, set the authentication absolute timeout to 1 second. This workaround opens a 1-second window of opportunity that might allow non-authenticated users to go through the firewall if they are coming from the same source IP address.
- **Authentication inactivity**—The idle time until the authentication cache times out and users have to reauthenticate a new connection. This duration must be shorter than the Translation Slot value. This timeout is disabled by default. This timer is used in cut-through proxy only, which is a AAA rule.
- **Translation Slot**—The idle time until a NAT translation slot is freed. This duration must be at least 1 minute. The default is 3 hours.
- **PAT Translation Slot** (8.4(3) and later, not including 8.5(1) and 8.6(1))—The idle time until a PAT translation slot is freed, between 0:0:30 and 0:5:0. The default is 30 seconds. You may want to increase the timeout if upstream routers reject new connections using a freed PAT port because the previous connection might still be open on the upstream device.
- **Connection Holddown**—How long the system should maintain a connection when the route used by the connection no longer exists or is inactive. If the route does not become active within this holddown period, the connection is freed. The purpose of the connection holddown timer is to reduce the effect of route flapping, where routes might come up and go down quickly. You can reduce the holddown timer to make route convergence happen more quickly. The default is 15 seconds, the range is 00:00:00 to 00:00:15.

Step 3 Click **Apply**.

Protect Servers from a SYN Flood DoS Attack (TCP Intercept)

A SYN-flooding denial of service (DoS) attack occurs when an attacker sends a series of SYN packets to a host. These packets usually originate from spoofed IP addresses. The constant flood of SYN packets keeps the server SYN queue full, which prevents it from servicing connection requests from legitimate users.

You can limit the number of embryonic connections to help prevent SYN flooding attacks. An embryonic connection is a connection request that has not finished the necessary handshake between source and destination.

When the embryonic connection threshold of a connection is crossed, the ASA acts as a proxy for the server and generates a SYN-ACK response to the client SYN request using the SYN cookie method (see Wikipedia for details on SYN cookies). When the ASA receives an ACK back from the client, it can then authenticate that the client is real and allow the connection to the server. The component that performs the proxy is called TCP Intercept.

The end-to-end process for protecting a server from a SYN flood attack involves setting connection limits, enabling TCP Intercept statistics, and then monitoring the results.

Before you begin

- Ensure that you set the embryonic connection limit lower than the TCP SYN backlog queue on the server that you want to protect. Otherwise, valid clients can no longer access the server during a SYN attack. To determine reasonable values for embryonic limits, carefully analyze the capacity of the server, the network, and server usage.
- Depending on the number of CPU cores on your ASA model, the maximum concurrent and embryonic connections can exceed the configured numbers due to the way each core manages connections. In the worst case scenario, the ASA allows up to *n*-1 extra connections and embryonic connections, where *n* is the number of cores. For example, if your model has 4 cores, if you configure 6 concurrent connections and 4 embryonic connections, you could have an additional 3 of each type. To determine the number of cores for your model, enter the **show cpu core** command.

Procedure

Step 1 Choose **Configuration** > **Firewall** > **Service Policy**.

Step 2 Click **Add** > **Add Service Policy Rule**.

Alternatively, if you already have a rule for the servers you want to protect, edit the rule.

Step 3 Select whether to apply the rule to a specific interface or globally to all interfaces, and click **Next**.

Step 4 For Traffic Classification, select **Source and Destination IP Addresses (uses ACL)** and click **Next**.

Step 5 For the ACL rule, enter the IP addresses of the servers in **Destination**, and specify the protocol for the servers. Typically, you would use **any** for the **Source**. Click **Next** when finished.

For example, if you want to protect the web servers 10.1.1.5 and 10.1.1.6, enter:

- Source = any
- Destination = 10.1.1.5, 10.1.1.6
- Destination Protocol = tcp/http

Step 6 On the Rule Actions page, click the **Connection Settings** tab and fill in these options:

- **Embryonic Connections**—The maximum number of embryonic TCP connections per host up to 2000000. The default is **0**, which means the maximum embryonic connections are allowed. For example, you could set this to 1000.
- **Per Client Embryonic Connections**—The maximum number of simultaneous embryonic TCP connections for each client up to 2000000. When a new TCP connection is requested by a client that already has the maximum per-client number of embryonic connections open through the ASA, the ASA prevents the connection. For example, you could set this to 50.

Step 7 Click **Finish** to save the rule, and **Apply** to update the device.

Step 8 Choose **Configuration** > **Firewall** > **Threat Detection**, and enable at least the **TCP Intercept** statistics under the Threat Detection Statistics group.

You can simply enable all statistics, or just enable TCP Intercept. You can also adjust the monitoring window and rates.

Step 9 Choose **Home** > **Firewall Dashboard**, and look at the **Top Ten Protected Servers under SYN Attack** dashboard to monitor the results.

Click the **Detail** button to show history sampling data. The ASA samples the number of attacks 30 times during the rate interval, so for the default 30 minute period, statistics are collected every 60 seconds.

You can clear the statistics by entering the **clear threat-detection statistics tcp-intercept** command using **Tools > Command Line Interface.**

Customize Abnormal TCP Packet Handling (TCP Maps, TCP Normalizer)

The TCP Normalizer identifies abnormal packets that the ASA can act on when they are detected; for example, the ASA can allow, drop, or clear the packets. TCP normalization helps protect the ASA from attacks. TCP normalization is always enabled, but you can customize how some features behave.

To customize the TCP normalizer, first define the settings using a TCP map. Then, you can apply the map to selected traffic classes using service policies.

Procedure

Step 1 Choose **Configuration** > **Firewall** > **Objects** > **TCP Maps**.

Step 2 Do one of the following:

- Click **Add** to add a new TCP map. Enter a name for the map.
- Select a map and click **Edit**.

Step 3 In the **Queue Limit** field, enter the maximum number of out-of-order packets that can be buffered and put in order for a TCP connection, between 0 and 250 packets.

The default is 0, which means this setting is disabled and the default system queue limit is used depending on the type of traffic:

- Connections for application inspection and TCP check-retransmission have a queue limit of 3 packets. If the ASA receives a TCP packet with a different window size, then the queue limit is dynamically changed to match the advertised setting.
- For other TCP connections, out-of-order packets are passed through untouched.

If you set the Queue Limit to be 1 or above, then the number of out-of-order packets allowed for all TCP traffic matches this setting. For example, for application inspection and TCP check-retransmission traffic, any advertised settings from TCP packets are ignored in favor of the Queue Limit setting. For other TCP traffic, out-of-order packets are now buffered and put in order instead of passed through untouched.

Step 4 In the **Timeout** field, set the maximum amount of time that out-of-order packets can remain in the buffer, between 1 and 20 seconds.

If they are not put in order and passed on within the timeout period, then they are dropped. The default is 4 seconds. You cannot change the timeout for any traffic if the Queue Limit is set to 0; you need to set the limit to be 1 or above for the Timeout to take effect.

Step 5 For **Reserved Bits**, select how to handle packets that have reserved bits in the TCP header: **Clear and allow** (remove the bits before allowing the packet), **Allow only** (do not change the bits, the default), or **Drop** the packet.

Step 6 Select any of the following options:

- **Clear urgent flag**—Clears the URG flag in a packet before allowing it. The URG flag is used to indicate that the packet contains information that is of higher priority than other data within the stream. The TCP RFC is vague about the exact interpretation of the URG flag, therefore end systems handle urgent offsets in different ways, which may make the end system vulnerable to attacks.
- **Drop connection on window variation**—Drops a connection that has changed its window size unexpectedly. The window size mechanism allows TCP to advertise a large window and to subsequently advertise a much smaller window without having accepted too much data. From the TCP specification, “shrinking the window” is strongly discouraged.
- **Drop packets that exceed maximum segment size**—Drops packets that exceed the MSS set by the peer.
- **Check if transmitted data is the same as original**—Enables the retransmit data checks, which prevent inconsistent TCP retransmissions.
- **Drop packets which have past-window sequence**—Drops packets that have past-window sequence numbers, namely the sequence number of a received TCP packet is greater than the right edge of the TCP receiving window. To allow these packets, deselect this option and set the **Queue Limit** to 0 (disabling the queue limit).
- **Drop SYN Packets with data**—Drops SYN packets that contain data.
- **Enable TTL Evasion Protection**—Have the maximum TTL for a connection be determined by the TTL in the initial packet. The TTL for subsequent packets can decrease, but it cannot increase. The system will reset the TTL to the lowest previously-seen TTL for that connection. This protects against TTL evasion attacks.

  For example, an attacker can send a packet that passes policy with a very short TTL. When the TTL goes to zero, a router between the ASA and the endpoint drops the packet. It is at this point that the attacker can send a malicious packet with a long TTL that appears to the ASA to be a retransmission and is passed. To the endpoint host, however, it is the first packet that has been received by the attacker. In this case, an attacker is able to succeed without security preventing the attack.

- **Verify TCP Checksum**—Verifies the TCP checksum, dropping packets that fail verification.
- **Drop SYNACK Packets with data**—Drops TCP SYNACK packets that contain data.
- **Drop packets with invalid ACK**—Drops packets with an invalid ACK. You might see invalid ACKs in the following instances:
  - In the TCP connection SYN-ACK-received status, if the ACK number of a received TCP packet is not exactly same as the sequence number of the next TCP packet sending out, it is an invalid ACK.
  - Whenever the ACK number of a received TCP packet is greater than the sequence number of the next TCP packet sending out, it is an invalid ACK.

Note TCP packets with an invalid ACK are automatically allowed for WAAS connections.

Step 7 Set the action for packets that contain TCP options. You can clear the options before allowing the packets, or allow the packets without change. The default is to allow the three named options, while clearing all other options. Note that if a TCP connection is inspected, all options are cleared except the MSS and selective-acknowledgment (SACK) options, regardless of your configuration.

- **Clear Selective Ack**—Clears the selective acknowledgment mechanism option.
- **Clear TCP Timestamp**—Clears the TCP timestamp. Clearing the timestamp option disables PAWS and RTT.
- **Clear Window Scale**—Clears the window scale mechanism option.
- **Range**—Sets the action for unnamed options. The ranges can be within 6-7 and 9-255. To configure an action for a single option, enter the same number for the lower and upper range. Choose **Allow**, **Clear** or **Drop** (the packet) for the action, and click **Add** to add it to the list. To remove a configured range, select it and click **Delete**.

Step 8 Click **OK** and **Apply**.

You can now use the TCP map in a service policy. The map affects traffic only when applied through a service policy.

Step 9 Apply the TCP map to a traffic class using a service policy.

a) Choose **Configuration > Firewall > Service Policy Rules**.
b) Add or edit a rule. You can apply the rule globally or to an interface. For example, to customize abnormal packet handling for all traffic, create a global rule that matches any traffic. Proceed to the Rule Actions page.
c) Click the **Connection Settings** tab.
d) Choose **Use TCP Map** and select the map you created.
e) Click **Finish** or **OK**, then click **Apply**.

Bypass TCP State Checks for Asynchronous Routing (TCP State Bypass)

If you have an asynchronous routing environment in your network, where the outbound and inbound flow for a given connection can go through two different ASA devices, you need to implement TCP State Bypass on the affected traffic.

However, TCP State Bypass weakens the security of your network, so you should apply bypass on very specific, limited traffic classes.

The following topics explain the problem and solution in more detail.

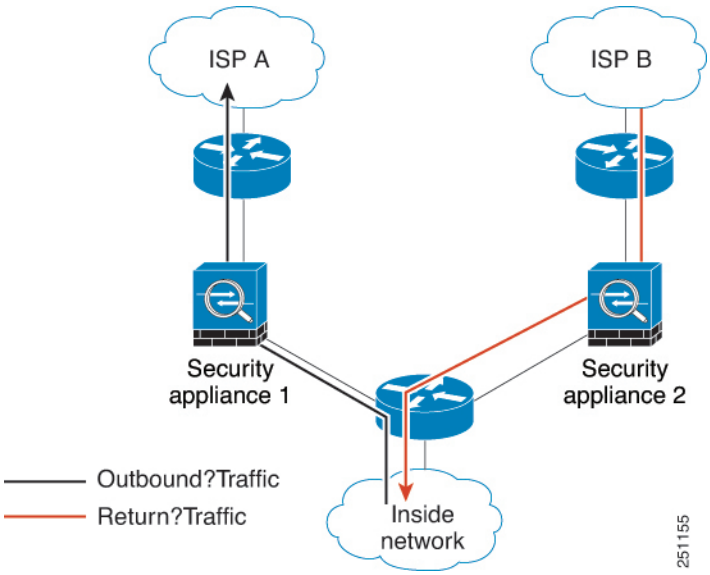
The Asynchronous Routing Problem

By default, all traffic that goes through the ASA is inspected using the Adaptive Security Algorithm and is either allowed through or dropped based on the security policy. The ASA maximizes the firewall performance by checking the state of each packet (new connection or established connection) and assigning it to either the session management path (a new connection SYN packet), the fast path (an established connection), or the control plane path (advanced inspection).

TCP packets that match existing connections in the fast path can pass through the ASA without rechecking every aspect of the security policy. This feature maximizes performance. However, the method of establishing the session in the fast path using the SYN packet, and the checks that occur in the fast path (such as TCP sequence number), can stand in the way of asymmetrical routing solutions: both the outbound and inbound flow of a connection must pass through the same ASA.

For example, a new connection goes to Security Appliance 1. The SYN packet goes through the session management path, and an entry for the connection is added to the fast path table. If subsequent packets of this connection go through Security Appliance 1, then the packets match the entry in the fast path, and are passed through. But if subsequent packets go to Security Appliance 2, where there was not a SYN packet that went through the session management path, then there is no entry in the fast path for the connection, and the packets are dropped. The following figure shows an asymmetric routing example where the outbound traffic goes through a different ASA than the inbound traffic:

Figure 1: Asymmetric Routing



If you have asymmetric routing configured on upstream routers, and traffic alternates between two ASA devices, then you can configure TCP state bypass for specific traffic. TCP state bypass alters the way sessions are established in the fast path and disables the fast path checks. This feature treats TCP traffic much as it treats a UDP connection: when a non-SYN packet matching the specified networks enters the ASA, and there is not a fast path entry, then the packet goes through the session management path to establish the connection in the fast path. Once in the fast path, the traffic bypasses the fast path checks.

Guidelines and Limitations for TCP State Bypass

TCP State Bypass Unsupported Features

The following features are not supported when you use TCP state bypass:

- Application inspection—Inspection requires both inbound and outbound traffic to go through the same ASA, so inspection is not applied to TCP state bypass traffic.
- AAA authenticated sessions—When a user authenticates with one ASA, traffic returning via the other ASA will be denied because the user did not authenticate with that ASA.
- TCP Intercept, maximum embryonic connection limit, TCP sequence number randomization—The ASA does not keep track of the state of the connection, so these features are not applied.
- TCP normalization—The TCP normalizer is disabled.
- Service module functionality—You cannot use TCP state bypass and any application running on any type of service module, such as ASA FirePOWER.
- Stateful failover.

TCP State Bypass NAT Guidelines

Because the translation session is established separately for each ASA, be sure to configure static NAT on both devices for TCP state bypass traffic. If you use dynamic NAT, the address chosen for the session on Device 1 will differ from the address chosen for the session on Device 2.

Configure TCP State Bypass

To bypass TCP state checking in asynchronous routing environments, carefully define a traffic class that applies to the affected hosts or networks only, then enable TCP State Bypass on the traffic class using a service policy. Because bypass reduces the security of the network, limit its application as much as possible.

Procedure

Step 1 Choose **Configuration** > **Firewall** > **Service Policy**.

Step 2 Click **Add** > **Add Service Policy Rule**.

Alternatively, if you already have a rule for the hosts, edit the rule.

Step 3 Select whether to apply the rule to a specific interface or globally to all interfaces, and click **Next**.

Step 4 For Traffic Classification, select **Source and Destination IP Addresses (uses ACL)** and click **Next**.

Step 5 For the ACL rule, enter the IP addresses of the hosts on each end of the route in **Source** and **Destination**, and specify the protocol as TCP. Click **Next** when finished.

For example, if you want to bypass TCP state checking between 10.1.1.1 and 10.2.2.2, enter:

- Source = 10.1.1.1
- Destination = 10.2.2.2
- Destination Protocol = tcp

Step 6 On the Rule Actions page, click the **Connection Settings** tab and select **TCP State Bypass**.

Step 7 Click **Finish** to save the rule, and **Apply** to update the device.

Disable TCP Sequence Randomization

Each TCP connection has two ISNs: one generated by the client and one generated by the server. The ASA randomizes the ISN of the TCP SYN passing in both the inbound and outbound directions.

Randomizing the ISN of the protected host prevents an attacker from predicting the next ISN for a new connection and potentially hijacking the new session.

You can disable TCP initial sequence number randomization if necessary, for example, because data is getting scrambled. For example:

- If another in-line firewall is also randomizing the initial sequence numbers, there is no need for both firewalls to be performing this action, even though this action does not affect the traffic.
- If you use eBGP multi-hop through the ASA, and the eBGP peers are using MD5. Randomization breaks the MD5 checksum.
- You use a WAAS device that requires the ASA not to randomize the sequence numbers of connections.
- You enable hardware bypass for the ISA 3000, and TCP connections are dropped when the ISA 3000 is no longer part of the data path.

Note We do not recommend disabling TCP sequence randomization when using clustering. There is a small chance that some TCP sessions won't be established, because the SYN/ACK packet might be dropped.

Procedure

Step 1 Choose **Configuration** > **Firewall** > **Service Policy**.

Step 2 Click **Add** > **Add Service Policy Rule**.

Alternatively, if you already have a rule for the targeted traffic, edit the rule.

Step 3 Select whether to apply the rule to a specific interface or globally to all interfaces, and click **Next**.

Step 4 For Traffic Classification, identity the type of traffic match. The class match should be for TCP traffic; you can identify specific hosts (with an ACL) do a TCP port match, or simply match any traffic. Click **Next** and configure the hosts in the ACL or define the ports, and click **Next** again.

For example, if you want to disable TCP sequence number randomization for all TCP traffic directed at 10.2.2.2, enter:

- Source = any
- Destination = 10.2.2.2
- Destination Protocol = tcp

Step 5 On the Rule Actions page, click the **Connection Settings** tab and uncheck **Randomize Sequence Number**.

Step 6 Click **Finish** to save the rule, and **Apply** to update the device.

Configure Connection Settings for Specific Traffic Classes (All Services)

You can configure different connection settings for specific traffic classes using service policies. Use service policies to:

- Customize connection limits and timeouts used to protect against DoS and SYN-flooding attacks.
- Implement Dead Connection Detection so that valid but idle connections remain alive.
- Disable TCP sequence number randomization in cases where you do not need it.
- Customize how the TCP Normalizer protects against abnormal TCP packets.
- Implement TCP State Bypass for traffic subject to asynchronous routing. Bypass traffic is not subject to inspection.
- Decrement time-to-live (TTL) on packets so that the ASA will show up on trace route output.

Note If you decrement time to live, packets with a TTL of 1 will be dropped, but a connection will be opened for the session on the assumption that the connection might contain packets with a greater TTL. Note that some packets, such as OSPF hello packets, are sent with TTL = 1, so decrementing time to live can have unexpected consequences for transparent mode ASA devices. The decrement time-to-live settings does not impact the OSPF process when ASA is operating in a routed mode.

You can configure any combination of these settings for a given traffic class, except for TCP State Bypass and TCP Normalizer customization, which are mutually exclusive.

Tip This procedure shows a service policy for traffic that goes through the ASA. You can also configure the connection maximum and embryonic connection maximum for management (to the box) traffic.

Before you begin

If you want to customize the TCP Normalizer, create the required TCP Map before proceeding.

Procedure

Step 1 Choose **Configuration** > **Firewall** > **Service Policy**, and open a rule.

- To create a new rule, click **Add > Add Service Policy Rule**. Proceed through the wizard to the Rules page.
- If you have a rule for which you are changing connection settings, select it and click **Edit**.

Step 2 On the Rule Actions wizard page or tab, select the **Connection Settings** tab.

Step 3 To set maximum connections, configure the following values in the Maximum Connections area:

By default, there are no connection limits. If you implement limits, the system must start tracking them, which can increase CPU and memory usage and result in operational problems for systems under heavy load, especially in a cluster.

- **Maximum TCP & UDP Connections**—(TCP, UDP.) The maximum number of simultaneous connections for all clients in the traffic class, up to 2000000. The default is 0, which means the maximum possible connections are allowed. For TCP connections, this applies to established connections only.
- **Embryonic Connections**—Specifies the maximum number of embryonic TCP connections per host up to 2000000. An embryonic connection is a connection request that has not finished the necessary handshake between source and destination. The default is **0**, which means the maximum embryonic connections are allowed. By setting a non-zero limit, you enable TCP Intercept, which protects inside systems from a DoS attack perpetrated by flooding an interface with TCP SYN packets. Also set the per-client options to protect against SYN flooding.
- **Per Client Connections**—(TCP, UDP.) Specifies the maximum number of simultaneous connections for each client up to 2000000. When a new connection is attempted by a client that already has opened the maximum per-client number of connections, the ASA rejects the connection and drops the packet. For TCP connections, this includes established, half-open, and half-closed connections.
- **Per Client Embryonic Connections**—Specifies the maximum number of simultaneous TCP embryonic connections for each client up to 2000000. When a new TCP connection is requested by a client that already has the maximum per-client number of embryonic connections open through the ASA, the ASA prevents the connection.

Step 4 To configure connection timeouts, configure the following values in the TCP Timeout area:

- **Embryonic Connection Timeout**—The idle time until an embryonic (half-open) TCP connection slot is freed. Enter 0:0:0 to disable timeout for the connection. The default is 30 seconds.
- **Half Closed Connection Timeout**—The idle timeout period until a half-closed connection is closed, between 0:5:0 (for 9.1(1) and earlier) or 0:0:30 (for 9.1(2) and later) and 1193:0:0. The default is 0:10:0. Half-closed connections are not affected by DCD. Also, the ASA does not send a reset when taking down half-closed connections.
- **Idle Connection Timeout**—The idle time until a connection slot (of *any* protocol, not just TCP) is freed. Enter 0:0:0 to disable timeout for the connection. This duration must be at least 5 minutes. The default is 1 hour.
- **Send reset to TCP endpoints before timeout**—Whether the ASA should send a TCP reset message to the endpoints of the connection before freeing the connection slot.
- **Dead Connection Detection (DCD)**—Whether to enable Dead Connection Detection (DCD). Before expiring an idle connection, the ASA probes the end hosts to determine if the connection is valid. If both hosts respond, the connection is preserved, otherwise the connection is freed. Set the maximum number of retries (default is 5, the range is 1-255) and the retry interval, which is the period to wait after each unresponsive DCD probe before sending another probe (0:0:1 to 24:0:0, default is 0:0:15). When operating in transparent firewall mode, you must configure static routes for the endpoints. You cannot use DCD in a cluster.

  For systems that are operating in a high-availability configuration, we recommend that you do not set the interval to less than one minute (0:1:0). If the connection needs to be moved between systems, the

changes required take longer than 30 seconds, and the connection might be deleted before the change is accomplished.

Step 5 To disable randomized sequence numbers, uncheck **Randomize Sequence Number**.

Randomizing the ISN of the protected host prevents an attacker from predicting the next ISN for a new connection and potentially hijacking the new session.

Step 6 To customize TCP Normalizer behavior, check **Use TCP Map** and choose an existing TCP map from the drop-down list (if available), or add a new one by clicking **New**.

Step 7 To decrement time-to-live (TTL) on packets that match the class, check **Decrement time to live for a connection**.

Decrementing TTL is necessary for the ASA to show up in trace routes as one of the hops. You must also increase the rate limit for ICMP Unreachable messages on **Configuration > Device Management > Management Access > ICMP**.

Step 8 To enable TCP state bypass, check **TCP State Bypass**.

Step 9 Click **OK** or **Finish**.

Monitoring Connections

Use the following pages to monitor connections:

- **Home** > **Firewall Dashboard**, and look at the **Top Ten Protected Servers under SYN Attack** dashboard to monitor TCP Intercept. Click the **Detail** button to show history sampling data. The ASA samples the number of attacks 30 times during the rate interval, so for the default 30 minute period, statistics are collected every 60 seconds.
- **Monitoring** > **Properties** > **Connections**, to see current connections.
- **Monitoring** > **Properties** > **Connection Graphs**, to monitor performance.

In addition, you can enter the following commands using **Tools** > **Command Line Interface**.

- **show conn** [**detail**]

  Shows connection information. Detailed information uses flags to indicate special connection characteristics. For example, the “b” flag indicates traffic subject to TCP State Bypass.

- **show service-policy**

  Shows service policy statistics, including Dead Connection Detection (DCD) statistics.

- **show threat-detection statistics top tcp-intercept** [**all** | **detail**]

  View the top 10 protected servers under attack. The **all** keyword shows the history data of all the traced servers. The **detail** keyword shows history sampling data. The ASA samples the number of attacks 30 times during the rate interval, so for the default 30 minute period, statistics are collected every 60 seconds.

History for Connection Settings

Feature Name	Platform Releases	Description
TCP state bypass	8.2(1)	This feature was introduced. The following command was introduced: **set connection advanced-options tcp-state-bypass**.
Connection timeout for all protocols	8.2(2)	The idle timeout was changed to apply to all protocols, not just TCP. The following screen was modified: Configuration > Firewall > Service Policies > Rule Actions > Connection Settings.
Timeout for connections using a backup static route	8.2(5)/8.4(2)	When multiple static routes exist to a network with different metrics, the ASA uses the one with the best metric at the time of connection creation. If a better route becomes available, then this timeout lets connections be closed so a connection can be reestablished to use the better route. The default is 0 (the connection never times out). To take advantage of this feature, change the timeout to a new value. We modified the following screen: Configuration > Firewall > Advanced > Global Timeouts.
Configurable timeout for PAT xlate	8.4(3)	When a PAT xlate times out (by default after 30 seconds), and the ASA reuses the port for a new translation, some upstream routers might reject the new connection because the previous connection might still be open on the upstream device. The PAT xlate timeout is now configurable, to a value between 30 seconds and 5 minutes. We modified the following screen: Configuration > Firewall > Advanced > Global Timeouts. *This feature is not available in 8.5(1) or 8.6(1).*
Increased maximum connection limits for service policy rules	9.0(1)	The maximum number of connections for service policy rules was increased from 65535 to 2000000. We modified the following screen: Configuration > Firewall > Service Policy Rules > Connection Settings.
Decreased the half-closed timeout minimum value to 30 seconds	9.1(2)	The half-closed timeout minimum value for both the global timeout and connection timeout was lowered from 5 minutes to 30 seconds to provide better DoS protection. We modified the following screens: Configuration > Firewall > Service Policy Rules > Connection Settings; Configuration > Firewall > Advanced > Global Timeouts.

Feature Name	Platform Releases	Description
Connection holddown timeout for route convergence.	9.4(3)	You can now configure how long the system should maintain a connection when the route used by the connection no longer exists or is inactive. If the route does not become active within this holddown period, the connection is freed. You can reduce the holddown timer to make route convergence happen more quickly. However, the 15 second default is appropriate for most networks to prevent route flapping. We modified the following screen: **Configuration** > **Firewall** > **Advanced** > **Global Timeouts**.