61200860L1-29.1G
March 2016

Configuration Guide

IPv4 Firewall Protection in AOS

This configuration guide is designed to provide you with an understanding of the Internet Protocol version 4 (IPv4) firewall protection provided by your ADTRAN Operating System (AOS) product. Included in this guide is an overview describing the types of IPv4 firewall protection, detailed command descriptions, and several network examples to cover a variety of firewall applications. The Troubleshooting section provides methods to identify and verify problems with your configuration.

This guide provides IPv4 firewall configuration instructions using the command line interface (CLI) and the graphical user interface (GUI). In addition, troubleshooting information using the CLI is included. This guide consists of the following sections:

Overview

A firewall is a collection of components configured to enforce specific access control parameters between your internal (trusted) network and any other (untrusted) network, such as the Internet. A firewall filters inbound and outbound packets to ensure that only authorized packets pass. The firewall can filter packets at several Open Systems Interconnection (OSI) levels and can enforce complex, customized policies.

This guide explains the firewall components in detail, as well as how to configure your AOS product for firewall protection in several different network scenarios. The following sections provide additional information about the components used in enforcing your firewall protection:

- Protection from known network attacks (refer to *Attack Checking on page 2*)
- Static filters (refer to *Static Filters on page 3*)
- Stateful inspection firewall (refer to *Stateful Inspection on page 4*)
- Stateless processing (refer to *Stateless Processing on page 9*)
- Allowing specific application traffic (refer to *Application-Specific Processing on page 10*)
- Network traffic management when used in conjunction with access control lists (ACLs) and access control policies (ACPs) (refer to *Access Control Lists and Access Control Policies on page 11*)

Attack Checking

Through attack checking, the IPv4 firewall detects and discards traffic that matches profiles of known networking exploits or attacks. Using the **ip firewall** command to enable firewall attack protection automatically detects and blocks these attacks. Some attack checks can be manually disabled, while other attack checks are enabled any time the firewall is enabled. Attack checks that can be enabled and disabled include: SYN flooding, WinNuke, Transmission Control Protocol (TCP) reset sequence number checking, reflexive traffic, and IP spoofing. Refer to *Enabling and Disabling Attack Checking on page 28* for more information on enabling or disabling these attack checks.

If the AOS device detects an attack, an attack log message will appear on the unit. Refer to *Appendix A. Attack Log Messages on page 87* for more information on attack log messages.

Table 1 outlines the types of traffic discarded by the firewall. Since many attacks use similar invalid traffic patterns, attacks other than the examples listed in the table could also be blocked by the firewall.

Table 1. Traffic Blocked by IPv4 Firewall Attack Protection Engine

Invalid Traffic Pattern	AOS IPv4 Firewall Response	Common Attacks
Larger than allowed packets	Any packets that are larger than those defined by standards will be dropped.	Ping of Death
Fragmented IP packets that produce errors when attempting to reassemble	The firewall intercepts all fragments for an IP packet and attempts to reassemble them before forwarding to the destination. If any problems or errors are found during reassembly, the fragments are dropped.	SynDrop, TearDrop, OpenTear, Nestea, Targa, Newtear, Bonk, Boink
Smurf Attack	The firewall drops any Internet Control Message Protocol (ICMP) ping and User Datagram Protocol (UDP) echo responses that are not part of an active session.	Smurf Attack

Table 1. Traffic Blocked by IPv4 Firewall Attack Protection Engine *(Continued)*

Invalid Traffic Pattern	AOS IPv4 Firewall Response	Common Attacks
IP Spoofing	The firewall drops any packets with a source IPv4 address that appears to be spoofed. The IPv4 route table is used to determine if a path to the source address is known (out of the IP interface from which the packet was received). For example, if a packet with a source IPv4 address of 10.10.10.1 is received on interface FR 1.16 and no route to 10.10.10.1 (through interface FR 1.16) exists in the route table, the packet is dropped. Traffic that bypasses spoofing checks includes packets from the router itself, Dynamic Host Configuration Protocol (DHCP) traffic, multicast and routing protocol traffic, and Virtual Router Redundancy Protocol (VRRP) traffic. Spoofing detection can be turned off on individual Access Control Policies (ACPs) to allow policy-based routing (PBR) or for any other case in which it would drop traffic that should not be dropped.	IP Spoofing
ICMP Control Message Floods and Attacks	The following types of ICMP packets are allowed through the firewall: echo, echo-reply, timestamp, timestamp reply, time to live (TTL) expired, dest unreachable, and quench. These ICMP messages are only allowed if they appear to be in response to a valid session. All others are discarded.	Twinge
Attacks that send TCP URG packets	Any TCP packets that have the urgent (URG) flag set are discarded by the firewall.	WinNuke, TCP XMAS Scan
Falsified IP Header Attacks	The firewall verifies that the packet's actual length matches the length indicated in the IPv4 header. If it does not, the packet is dropped.	Jolt/Jolt2
Land Attack	Any packets with the same source and destination IPv4 addresses are discarded.	Land Attack
Broadcast Source IP	Packets with a broadcast source IPv4 address are discarded.	
Invalid TCP Initiation Requests	Initial TCP synchronize (SYN) packets that have acknowledge (ACK), URG, reset (RST), or finished (FIN) flags set are discarded.	
Invalid TCP Segment Number RST	The sequence numbers for active TCP sessions are maintained in the firewall session database. If the firewall receives an RST segment with an unexpected (or invalid) sequence number, the packet is dropped.	
IP Source Route Option	All packets containing the IP source route option are dropped.	

Static Filters

Static filters are able to filter IP traffic using access control lists (ACLs) without the firewall enabled. As a result, the attack checks that are normally performed by the firewall are not affected. A separate filtering decision is made for every individual packet without regard to the states of the previous packets. In this regard, static filters are stateless. The static filter will either allow or discard a packet.

There are two types of static filters used by AOS:

- Access groups - These are applied to an interface and contain filter lists that can be set to filter either inbound or outbound traffic. To filter both inbound and outbound traffic, you must apply two access groups to the interface. Even if the same access group is to be used for both directions, it must be applied twice. Access groups can be used to filter any type of traffic to, from, or through the unit. The advantages of using access groups is they do not require the firewall to be enabled.
- Access classes - These are applied to specific servers that reside in the unit, such as the Telnet or Hypertext Transfer Protocol (HTTP) server. They filter login access to the unit for the specific protocol associated with the server. Standard ACLs are used as the filter for access classes.

When access groups are used, an inbound access group applied to the public interface typically needs to reject sessions initiated from the Internet while allowing responses to sessions initiated from the internal network. This can be difficult to accomplish because the filter lists have no knowledge of the status of the session (sequence numbers, inactivity time, etc.). As a result, it is possible that an attacker could fool the configured filter lists and direct malicious traffic through the firewall.

For example, consider an application where a host located behind a firewall device initiates an outbound session to a server on the Internet. If access groups are used, two must be defined and applied to interfaces in the following manner:

- An access group that allows outbound traffic from the host to the Internet must be applied inbound on the internal interface.
- An access group that allows inbound traffic (responses to the initiated session) from the Internet to the host must be applied inbound on the public interface.

Stateful Inspection

Contrary to using access groups with the firewall disabled, the IPv4 firewall, when enabled, performs stateful inspection and attack checks on traffic to and through the unit. Every interface has a single ACP, whether defined explicitly or using the default ACP, which inspects traffic entering that interface. Typically, an IPv4 ACP is configured on the internal interface that permits hosts to initiate traffic to the Internet, often by using network address translation (NAT). A second IPv4 ACP is configured on the public interface that discards traffic initiated from the Internet. The AOS stateful inspection firewall creates an association for a session initiated from the internal network and stores it in an internal database. When the server on the Internet sends a response back to the host, the AOS stateful inspection firewall recognizes that this traffic is associated with an allowed session and allows the traffic to pass through. Since the firewall has detailed knowledge about the current state of every session flowing through the device, it is much more difficult for an attacker to generate traffic that is not blocked by the firewall. Refer to *Access Control Policies on page 13* for more information on IPv4 ACPs and NAT.

Firewall Associations

Firewall associations are needed so that the AOS device does not have to perform the matching process for every packet that comes through the IPv4 firewall. The benefit of firewall associations is that they increase the speed of packet throughput. For example, if the AOS device is processing a long stream of packets that share the same protocol, source, and destination, and the same action should be performed on each packet, then rigorously matching every packet would be needlessly redundant and waste time. When the IPv4 firewall receives the first packet in a unique stream, it calculates the appropriate action to perform on the packet. Through the use of firewall associations, the AOS device can simply look at subsequent packets in a stream, note that the packets look the same as the first, and perform the calculated action upon them.

Active Policy Sessions

Active policy sessions are firewall associations that are currently alive, thus they allow faster processing of appropriate packets as described previously in *Firewall Associations on page 4*.

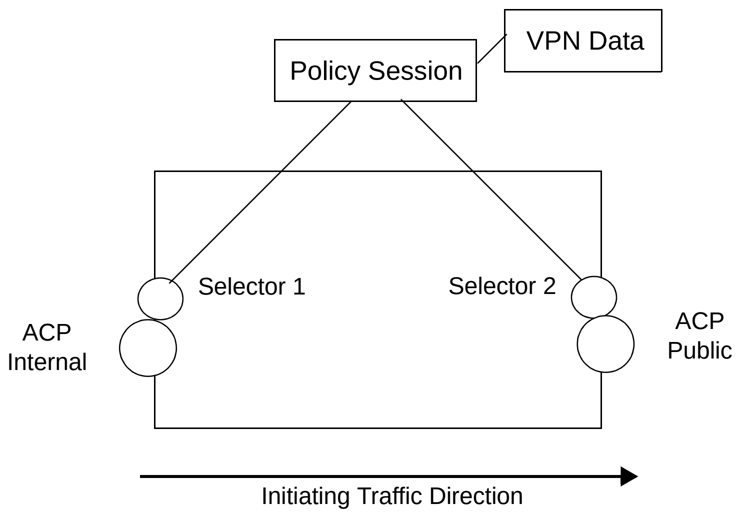


Figure 1. Architecture of a Policy Session

Figure 1 above shows a policy session that consists of two selectors, Selector 1 and Selector 2. Each selector is linked to an interface and the ACP applied to that interface. Selector 1 is associated with the original traffic attributes and Selector 2 is associated with the return traffic attributes. Selector 2 will be a mirror image of Selector 1 unless the traffic is processed by NAT.

Each selector contains the following information:

- Virtual routing and forwarding (VRF) ID (the VRF of the ingress interface on which the packet arrived)
- ACP ID
- Information on how a packet in this session should look when entering the interface for which the ACP is applied (due to NAT, some of the values could be different between the two selectors): protocol, source/destination IPv4 address, source/destination port.
- Source interface (the ingress interface)
- Destination interface(s) (the egress interface or interfaces if load sharing is enabled)

Each policy session has an assigned policy-timeout. Policy-timeouts specify the amount of time that should pass before policy sessions are deleted due to inactivity (for example, packets are not passing through the policy session). Timeouts are specified per protocol (TCP, UDP, ICMP, etc.), and, in the case of TCP and UDP, timeouts can be specified for a single port, a range of ports, or all ports. There is a limit of 10 configured policy-timeouts per protocol. The default timeout value for established TCP is 600 seconds and for all other protocols it is 60 seconds. Policy-timeouts override any server-specific login timeouts configured on the unit.

There are multiple ways to override some of the timeout settings:

- There are some conditions where the SIP application level gateway (ALG) overrides the timeouts for UDP 5060 and real-time transport protocol (RTP) traffic. If the SIP ALG decides that a policy session should be deleted quickly, the timeout value could be as low as 24 seconds. Otherwise, this value will be 60 seconds.
- The user can configure a TCP RST timeout and/or a TCP FIN timeout.

Creating Policy Sessions

There are several different circumstances that trigger the creation of a policy session:

1. A policy session is created when the first packet of a unique traffic flow arrives. Once the session is created, all other packets in the traffic flow will match the policy session’s 5-tuple selector and new sessions do not need to be created.
2. A policy session is created when traffic arrives that matches an existing passive association. A pending policy session (see *Pending Policy Sessions on page 8*) is similar to an active policy session but with some unknown fields. By inspecting the traffic, the AOS device can create an active policy session from the pending policy session by filling the unknown fields with the values that are now known.
3. Policy sessions are created for existing open local socket connections when the firewall is toggled from disabled to enabled. A local socket connection is a TCP connection, like Telnet or SSH, to the unit itself. A policy session is created for this type of connection so that traffic over the open socket is not disconnected from the firewall.

Reworking Policy Sessions

When a policy session is reworked, one or more of the important values in the association structure are changed. A rework is triggered by either a route table change or a change to an ACL in a PBR entry. Only active policy sessions are reworked.

A true rework is the most common type of rework. The only values in the selector that are changed in this type of rework are the ACP and the interface with which that selector is associated. A true rework simply replaces the ACP and/or interface on an association's selector on one end of the session with a different ACP and/or interface.

Fast NAT failover, a type of rework that is discussed in *Enabling Fast NAT Failover on page 33*, is used to enable the IPv4 firewall to perform failover on a policy session so that incorrect associations are not used. The per-packet load sharing non-NAT interface change rework is similar to the load sharing example described in *Enabling Fast NAT Failover on page 33* where fast NAT failover is necessary if the *sticky interface* goes down. The major difference with the per-packet load sharing non-NAT interface change rework is that the IPv4 firewall does not kill the policy session and then wait for the next matching packet to create a new policy session. Instead, the destination interface of the appropriate selector is changed.

Another type of rework is a virtual private network (VPN) Data Rework. *Figure 1 on page 5* shows that policy sessions include pointers to VPN data structures, which connect firewall associations with VPN security associations (SAs). When route table changes affect which crypto maps should be used by policy sessions, the policy sessions must be reworked so that they use the correct VPN SAs.

Deleting Active Policy Sessions

Different events can lead to the deletion of an active policy session. Some of these events are triggered manually, while others are triggered by another protocol or due to inactivity.

All currently active policy sessions can be manually deleted using the command **clear ip policy-sessions**. This command also contains optional parameters that allow the user to specify policy sessions limited to a particular VRF, only policy sessions related to a particular ACP, or a specific policy session. See *Table 8 on page 80* for more information on the **clear ip policy-sessions** command.

NOTE *Some ALGs (for example, SIP) will not allow policy sessions created by the ALG to be manually deleted.*

An active policy session will be deleted when the TTL, initially set to the policy timeout, decrements to zero. Refer to *Active Policy Sessions on page 5* for a detailed discussion on policy timeouts.

Active policy-sessions can be deleted by an ALG. There are many possible reasons why an ALG will delete a policy session. The SIP ALG, for example, can delete a policy session after receiving an ACK, after a Cancel, or after a failure case (such as if a registration fails). Active policy sessions can also be deleted by a SIP server. When the AOS device is acting like a SIP server, then deletion of a policy session can occur when a new policy session for SIP traffic needs to be made and stale SIP policy sessions need to be deleted.

When a VRF instance is deleted, then all active policy sessions using that VRF are also deleted. Another application where active policy sessions must be deleted is the transfer to a backup router when running VRRP. VRRP is a protocol where two or more physical routers advertise themselves as one virtual router, but only one of the physical routers performs the actual routing at any given time. If the currently active physical router in the group fails, then one of the other routers will immediately take over routing duties; thus increasing network connection reliability. However, when this transfer occurs, active policy sessions using the interface to which the failed physical router is connected must be deleted.

NOTE *There can be up to a 7-second delay before the policy session is actually deleted. Also, when a TCP policy session is deleted, the AOS device will send TCP RSTs to both ends of the session (as long as the session is not stateless and if the unit has not already seen a FIN for the session).*

Pending Policy Sessions

Pending policy sessions are like active policy sessions except that some of the fields are unknown. A pending policy session does not represent a currently active stream of traffic, but rather it represents potential traffic. The possible unknown fields include the source and/or destination IPv4 addresses and the source and/or destination ports. Pending policy sessions create a larger hole in the firewall because they are not as specific as active policy sessions in describing what type of traffic should be allowed to use the policy session. As a result, pending policy sessions have shorter lifetimes than active policy sessions.

Creating Pending Policy Sessions

An ALG creates a pending policy session when it expects future traffic from a source and/or destination of which the IPv4 firewall has partial knowledge. For example, if it is determined that traffic with destination address 10.10.10.1 and destination port 50 should be allowed to go through an ALG, then a pending policy session is created specifying those destination fields, but leaving the source address and source port fields as unknowns. This might be the case in an application where the customer would like to allow a public device to be able to access a service that resides on the internal side of the AOS device. Likewise, a pending policy session could also be created for the purpose of allowing unknown internal devices to send outgoing traffic.

Pending policy sessions are also created when SIP phones register themselves with a SIP server through the AOS device. After registration, the IPv4 firewall will know the IPv4 addresses (on both the public and internal sides) from which the AOS device can expect call-initiating SIP traffic.

Deleting Pending Policy Sessions

A pending policy session will be deleted when the TTL decrements to zero due to inactivity. Since the session was passive to begin with, this just means that the policy session was never matched. If traffic arrives that matches the known fields of the pending policy session, then the unknown fields are filled in and the pending policy session is changed into an active policy session.

Local Traffic Processing

The AOS IPv4 firewall can be configured to process local traffic only, that is, traffic arriving at the unit's local IP stack. When configured, routed traffic is allowed to flow through the AOS unit uninspected, but locally destined traffic is inspected by the firewall. This feature allows the firewall to protect local services running on the AOS unit even when routed traffic bypasses the firewall. When local traffic processing is enabled, several other security features are impacted, such as IPsec, policy classes, IP route cache, Generic Routing Encapsulation (GRE), and NAT.

- Local traffic only firewall processing cannot be used with cryptography (**ip crypto**) because for IPsec to function, traffic must proceed through the firewall. If the firewall is configured to process local traffic only, routed traffic that requires IPsec protection will not flow through the firewall and therefore will not receive IPsec protection.
- Policy classes are applied only to traffic destined to the local stack when local traffic processing is enabled. The **self** policy class is applied to local traffic originating from the local stack, allowing all traffic, and cannot be changed.
- IP route cache entries are not created for local destinations or for the Loopback interface when local traffic processing enabled.
- Local GRE traffic encapsulated by a GRE tunnel interface will bypass the firewall when local traffic processing is enabled.

- The full firewall is required any time NAT is needed to translate packets that would typically be forwarded by the AOS unit. The local firewall is not sufficient.

Stateless Processing

Stateless processing refers to when a packet is processed without full consideration being given to what traffic the unit has already seen. In other words, it doesn't care about the current state of the traffic flow; the packet is considered apart from previous packets (except that attack checks might be performed). As mentioned earlier, access group processing is always stateless. The following sections describe other cases of stateless processing when the firewall is enabled.

Stateless Processing That Creates Policy Sessions

In some cases, statelessly processed traffic creates active policy sessions. The traffic is not susceptible to firewall timeouts and does not go through an application-level gateway (ALG) since ALGs perform stateful processing. However, depending on the protocol, stateless traffic might undergo attack checks.

NOTE *Refer to Appendix A. Attack Log Messages on page 87 for a list of the attack checks that can be bypassed using a stateless IPv4 ACP entry.*

The following methods are used to specify traffic that should not undergo ALG or stateful attack checks, but for which a policy session should be created:

- Enable **ip crypto,** but leave **ip firewall** disabled. In this case, the **ip crypto** command allows VPN policy sessions and only applies the default and self ACP. (Traffic will be processed statelessly.)
- Add the keyword **stateless** to the end of an **allow** statement in an IPv4 ACP. Traffic that matches the **allow** statement will create a policy session, but will not undergo attack checking. This method is most commonly used to allow traffic between internal subnets that need to communicate with each other. For example, network 192.168.1.x communicating with network 192.168.2.x.

There are several types of traffic where it is helpful or necessary for a stateless policy session to be created:

- VPN traffic going to an interface with the default ACP, but **ip firewall** is disabled.
- Routing layer 4 protocol traffic that is not well known. This is traffic that is not TCP, UDP, ICMP, generic routing encapsulation (GRE), authentication header (AH), encapsulating security payload (ESP), internet group management protocol (IGMP), open shortest path first (OSPF), protocol independent multicast (PIM), or VRRP since this type of traffic must bypass attack checks or it will be dropped.
- Encrypted traffic not destined for the local router, that must be able to pass through the unit. This traffic could be dropped if it went through the firewall.
- Packets within a certain traffic flow that will use the same ACP as every other packet. A configuration in which the ingress and/or egress interfaces of different packets could be different (such as a remote unit load sharing the traffic back to the local router).

There are also a couple of methods for specifying traffic that should not go through an ALG, but should undergo attack checks and for which a policy session should be created:

- Add the keyword **no-alg** to the end of a **nat** statement in an IPv4 ACP. Traffic that matches the NAT statement will undergo attack checks because it is going through the firewall, but the **no-alg** keyword will ensure that it does not go through any ALG.
- Disable specific ALGs that you do not want to use to process traffic. It should be noted that in AOS the H.323 and Microsoft Service Network (MSN) ALGs are disabled by default and the SIP, Point-to-Point Tunneling Protocol (PPTP), File Transfer Protocol (FTP), and Internet relay chat (IRC) ALGs are enabled by default.

There are several types of traffic where it is helpful or necessary that a policy session is created and attack checks are enabled, but the traffic is not sent to an ALG for processing:

- Traffic on which NAT is not being performed at all. One of the main benefits to using ALGs is to allow certain application traffic to be handled correctly when NAT is being used. If NAT is not being used, turning off ALGs might be desired for faster processing.
- Traffic on which NAT is being performed, but is being handled entirely outside of the unit.
- An application not associated with an ALG on the unit uses a port that is registered with one of the ALGs. In this case, you don’t want the traffic generated by the application to be handled incorrectly, and must make sure that the traffic is not processed by the corresponding ALG. For example, non HTTP traffic destined to TCP port 80 should not be parsed by an HTTP ALG.
- To filter HTTP traffic from certain hosts, but not from others. For example, you might be using the HTTP uniform resource locator (URL) Filter ALG, but you could use the keyword **no-alg** on the ACP entries that matched the traffic from hosts for which you do not want to filter URL traffic.

Stateless Processing That Does Not Create Policy Sessions

For the following types of statelessly processed traffic, active policy sessions are not required because the traffic is implicitly allowed by AOS firewall:

- DHCP traffic to or from the unit (not through the unit). When the unit is a DHCP server, this can mean broadcast packets sent by a DHCP client seeking an IPv4 address. When the unit has a DHCP client interface of its own, this can mean unicast packets sent to it by a DHCP server.
- Multicast traffic, including OSPF, PIM, IGMP, Routing Information Protocol version 2 (RIPv2), and VRRP traffic. (Other types of multicast traffic are unrecognized by AOS.)
- Internet key exchange (IKE) traffic (using UDP ports 500 and 4500) that terminates in the unit.
- ESP and AH traffic that terminates in the unit.

Application-Specific Processing

Certain applications need special handling to work correctly in the presence of a firewall. AOS uses ALGs for these applications. ALGs are aware of protocols not easily integrated with NAT or firewalls, and create a policy session that allows these protocols to work transparently. For example, the FTP ALG creates policy sessions that allow the control session (using TCP Port 21) to pass data and also creates policy sessions that allow the server-initiated data sessions to work (using TCP Port 20) in active FTP mode. This allows FTP clients to pass through the AOS firewall and ACPs without using FTP passive mode.

Several purposes for using ALGs are:

- Allowing application-specific deep packet inspection (DPI), meaning that it is not just the header of an incoming packet that is inspected, but also the packet data. This information can help the ALG decide if the packet needs to be rerouted, modified, or dropped.
- Allowing application-specific NAT (or deep NAT), which translates the addresses within the application layer of a packet and not just in the packet header.
- Creating pending policy sessions for expected application-specific traffic, allowing the traffic to go through the unit without being dropped by the firewall.
- Increasing security by performing additional attack checks directly related to the type of traffic being handled by the ALG.

The AOS firewall includes ALGs for handling the following applications and protocols:

- America Online (AOL)
- AOL Instant Messenger (AIM®)
- CUseeme
- FTP
- H.323: H.245 Q.931 ASN1 PER decoding and encoding
- HTTP
- ICQ®
- IRC
- Layer 2 Tunneling Protocol (L2TP)
- Microsoft® Games
- Microsoft® Gaming Zone (MSZONE)
- MSN
- Net2Phone
- pcAnywhere™
- PPTP
- Quake®
- Real-Time Streaming Protocol (RTSP)
- SIP
- Structured Query Language (SQL)
- Trivial File Transfer Protocol (TFTP)
- VPN ALGs: ESP and IKE

Refer to *Enabling and Disabling Application-Level Gateways on page 29* for more information on enabling or disabling the ALGs.

Access Control Lists and Access Control Policies

ACLs and access control policies (ACPs) regulate traffic through the routed network. When designing your traffic flow configuration, it is important to keep the following in mind:

- An ACL serves as a packet selector, defining exactly which packets should take the given action.

- An ACP defines the action to take on the packets selected by the ACL.
- An ACL is inactive until it is assigned to an active ACP.
- An ACP is inactive until it is assigned to an interface and the firewall is enabled.

Figure 2 illustrates the steps necessary for activating ACLs and ACPs.

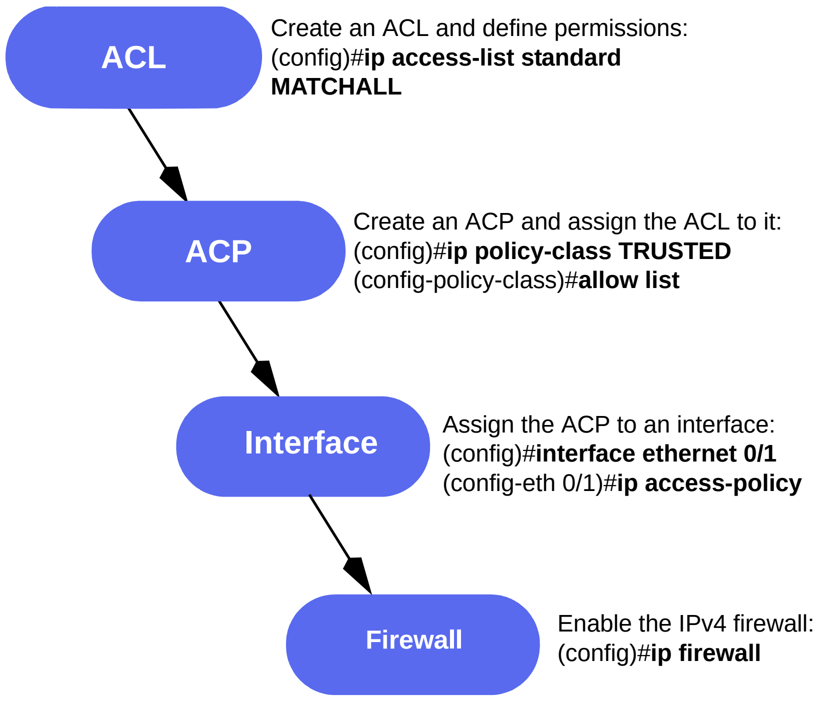

ACL
Create an ACL and define permissions:
(config)#**ip access-list standard MATCHALL**
ACP
Create an ACP and assign the ACL to it:
(config)#**ip policy-class TRUSTED**
(config-policy-class)#**allow list**
Interface
Assign the ACP to an interface:
(config)#**interface ethernet 0/1**
(config-eth 0/1)#**ip access-policy**
Firewall
Enable the IPv4 firewall:
(config)#**ip firewall**

Figure 2. Activating ACLs and ACPs

Access Control Lists

ACLs are used as packet selectors and are composed of an ordered list of entries. Each ACL entry begins with either a **permit** or **deny** keyword. A **permit** ACL is used to allow packets meeting the specified pattern to be processed by the feature using the ACL (in the case of the firewall, an ACP). A **deny** ACL causes packets meeting the specified pattern to advance to the next ACP entry.

When an ACL is being used to inspect an IP packet, the entries in the ACL are processed from top-to-bottom, in the order in which the entries were added to the ACL. If an IP packet does not match the criteria specified by a certain entry, then it is compared to the criteria in the next entry. If all of the entries in an ACL are examined and there is no match, there is an implicit **deny all** at the end of the ACL that causes the next rule in the ACP to be processed.

There are two types of ACLs, standard and extended.

- **Standard**
  The only field in the packet inspected by standard ACLs is the IPv4 source address. Packets sent from specific subnets or specific hosts can be specified as either **permit** or **deny**. The **any** keyword is used to specify either a **permit** or a **deny** of all packets. For standard ACLs (but not extended), an ACL cache is created to speed up the packet matching process.

- **Extended**
  There are a number of fields in the packet that can be inspected by extended ACLs. These include protocol, source and destination IPv4 addresses, source and destination ports, and most fields in the IP, TCP, UDP, and ICMP headers. The **any** keyword can be used in reference to the packet's source, destination, or both.

For more information on ACLs, refer to the configuration guide *IPv4 ACLs in AOS* (available online at https://supportforums.adtran.com).

NOTE *When entries exist within an ACL, an implicit deny statement is automatically placed at the end of the entries. When an ACL is empty (no entries), the ACL implicitly allows everything.*

Access Control Policies

ACPs are applied when IPv4 packets are received on an inbound interface (rather than outbound). Every inbound interface can have only one associated ACP. They are faster to process than access groups. Each IPv4 ACP entry consists of a selector (an ACL) and an action (allow, discard, or NAT). Also, an ACP entry can specify a destination ACP to match as part of its rule. When packets are received on an interface, the configured ACPs are applied to determine whether the data is processed or discarded. After inspecting traffic, the IPv4 ACP performs one of four actions:

- The **allow** keyword permits traffic through the firewall without changing the IP packet's source or destination IPv4 addresses or Layer 4 ports.
- The **discard** keyword drops the traffic.
- The **nat source** keyword translates the source IPv4 address to a specified IPv4 address (or to the primary IPv4 address of the specified interface) and creates an association in the firewall.
- The **nat destination** keyword translates the destination IPv4 address to a specified IPv4 address and creates an association in the firewall.

ACPs are order dependent. When an IPv4 packet is evaluated, the matching engine begins with the first entry in the list and progresses through the entries until it finds a match. The first entry that matches is executed. Typically, the most specific entries should be at the top and the more general at the bottom.

There are several types of ACPs, including:

- The **self** ACP handles traffic whose source or destination is the router itself. This ACP is unique and cannot be deleted.
- The **default** ACP is applied to every interface that does not have a user-defined ACP and allows all traffic while performing stateful checks with **ip firewall** enabled. It will allow all traffic while performing stateless checks if **ip crypto** is enabled while **ip firewall** is disabled. The default ACP is unique and cannot be deleted.
- The **user-defined** ACP is defined by the user. AOS supports up to 20 user-defined ACPs.

IPv4 ACP Entry Matching

As described previously, an IPv4 ACP entry can specify an **allow**, **discard**, **nat source**, or **nat destination** action. In order for an action to be taken on the IPv4 packet being inspected, it must first fit the match criteria. There are up to three steps taken when performing IPv4 packet matching on an ACP entry:

1. Check the optional destination ACP, if it exists. AOS reviews the IP packet's destination IPv4 address and uses the route table to determine what the IP packet's destination interface is for that address. If the ACP applied to that interface is the same as the ACP specified in the entry, the process continues to Step 2. Otherwise, the match fails and there is no need to perform Step 2.

NOTE *It is recommended that this Step 1 above always be used on ACP entries. It can be useful if the same ACL is used on multiple ACP entries within the same ACP. If load sharing is being used, this will differentiate traffic that is being sent out two different interfaces, each of which will have a different associated ACP.*

2. Check the optional NAT pool, if it exists. Refer to the configuration guide *NAT Pools in AOS* (available online at https://supportforums.adtran.com) for more information on NAT pools.
3. Check the ACL.

NOTE *If no entries in an IPv4 ACP match the IP packet, then the packet is dropped by the last entry, which contains an implicit* ***discard all****. Also, if an IPv4 ACP has been created, but contains no entries, then the IPv4 packet is dropped.*

Source NAT

Source NAT involves translating the source IPv4 address field of the packet into a different IPv4 address. There are several varieties of source NAT, including many-to-one and static one-to-one.

Many:1 NAT

Many:1 NAT, also called network address port translation (NAPT) or port address translation (PAT) is when the source IPv4 addresses of many different traffic flows are translated into one address. The user can either give a specific IPv4 address to replace the internal addresses or simply specify an interface, in which case the primary IPv4 address currently assigned to the interface will be the replacement IPv4 address. Since all flows are given the same source IPv4 address with NAPT, they will be given different source ports in order to differentiate them.

NOTE *The specified address for source NAT must be an address that has actually been assigned to the AOS device or any of its interfaces.*

Static 1:1 NAT

It is recommended that new static 1:1 NAT setups should be configured using static 1:1 NAT pools instead of the older configuration method discussed below. The following content is included for the benefit of those customers who are more familiar with the original means of configuring static 1:1 NAT in AOS. Refer to the configuration guide NAT Pools in AOS (available online at https://supportforums.adtran.com) *for more information on the recommended way to configure static 1:1 NAT.*

When Static 1:1 NAT is used, a packet entering the AOS device from one side of the unit that has a specific source IPv4 address, x.x.x.x, is translated to the source IPv4 address y.y.y.y. In other words, there is a one-to-one correspondence between the address to be translated and the address to which it will be translated. Also, to ensure that traffic initiated from the other side of the AOS device is sent outbound to the correct IPv4 address, destination NAT must be applied such that destination address y.y.y.y in a packet is translated to destination address x.x.x.x.

Some vendors require that when performing static one-to-one NAT, the keyword ***overload*** *be removed from the source NAT command (which is required in the source NAT command when performing many-to-one NAT). AOS does not currently support this;* ***overload*** *is always required in a source NAT command unless NAT pools are used to perform static 1:1 NAT. Refer to the configuration guide NAT Pools in AOS* (available online at https://supportforums.adtran.com) *for more information on NAT pools.*

Static one-to-one NAT requires source port preservation be enabled so that the port mappings are also one-to-one (Source port preservation is enabled by default). This type of NAT is helpful if you use an application for which AOS does not have an ALG supporting public-side initiated traffic. One of the purposes for ALGs is to create holes on the public side of the unit (through the use of pending policy sessions) so that specific traffic initiated from that side is allowed through on certain ports. Lack of such an ALG requires the presence of corresponding source NAT (on the private side) and destination NAT (on the public side) commands.

Improvements were made to the firewall functionality in AOS 17.1 to allow static 1:1 NAT to work with source port preservation. This allows the source ports of the traffic originating from the private IPv4 interface to be preserved through NAT to the Public IPv4 interface.

Using Port Maps

When using source NAT, every NAT address/VRF combination has an associated port map. A port map is a structure that keeps track of which ports are being used by policy sessions that use the source NAT-address on the VRF and how many times the ports are being used. The same port can be used separately by each of the protocols (TCP, UDP, ICMP, and GRE) and the same port can also be used by different port maps. Port maps are a necessary part of source port preservation, which is described next.

Source Port Preservation

Source port preservation is a feature that tries to retain the original source port when performing source NAT. This is a global feature and is either enabled or disabled for all VRFs. Source port preservation can operate either with or without recording the source IPv4 address.

- *With recorded source IPv4 address.*
  The packet's original source port is compared to the NAT ports being used on other policy sessions. If the port is not already being used, it is retained as the source port. If the port is already being used, AOS checks the original source IPv4 address of the policy session that is using the NAT port. If it is the same as the original source IPv4 address of the policy session AOS is trying to create, the source port is reused. Otherwise, AOS picks a new port number and tries again. This option uses more memory than *without recorded source address*.
- *Without recorded source address.*
  This is the default setting in AOS. First, the port map is used to compare the IP packet's original source port to the NAT ports being used on other policy sessions. If the port is not already being used, it is kept as the source port. Otherwise, AOS picks another port number and tries again. Port numbers 1025 and above are dynamically assigned.

Destination NAT

Destination NAT involves translating the destination IPv4 address field (and possibly the port number) of an IP packet. One situation in which destination NAT is especially helpful is if you want to provide public access to internal servers through port forwarding. In port forwarding, incoming traffic is differentiated from the public side of the unit by looking at the matching criteria specified in an ACL. AOS translates the destination IPv4 address of IP packets that match the ACL to a specific internal IPv4 address, which directs the IP packet to a particular host or server. The destination port of the IP packet can also be changed to differentiate service on the router from that of the server(s) behind the router. For example, traffic destined to port 999 on the public side of the unit can be translated to port 23 of an internal server so that a Telnet session with the server can be established.

NOTE

Refer to the tech note Port Forwarding in AOS (available online at https://supportforums.adtran.com) *for more information on port forwarding.*

Processing ACPs and ACLs

The logical flow of how IP packets are processed by ACPs is described below, followed by ACLs. This information is provided to gain understanding using ACPs and ACLs together in an IPv4 firewall configuration.

ACPs

If the interface on which a packet arrives references an ACP that has not been created in the configuration, then the packet is dropped. The packet will also be dropped if the ACP has been created but contains no entries. If entries exist, they are processed from top-to-bottom, as they appear in the running configuration. If none of the entries match the packet, then the packet is dropped by the last entry, which contains an implicit **discard all**.

Exceptions

- Duplicate entries are allowed.
- IKE messages (using UDP ports 500 and 4500) to or from the unit are implicitly allowed unless they are matched by an explicit entry, in which case the action specified by the entry is taken.
- ESP messages to or from the unit are also implicitly allowed.
- Decrypted traffic must be explicitly allowed.

ACLs

If an ACP entry references an ACL that has not been created in the configuration, then the packet is matched by the entry unless a destination ACP was also specified and has already denied the packet. If the referenced ACL exists but contains no entries (an empty ACL), then the packet is matched. If entries exist, entries are processed from top-to-bottom, which is the order in which they were entered.

Exceptions

- Duplicate entries are NOT allowed.
- The last entry is an implicit **deny any**, meaning the packet is not matched by the ACL.

Hardware and Software Requirements and Limitations

Firewall protection is available on the AOS data and voice products as outlined in the *Product Feature Matrix* available online at https://supportforums.adtran.com. Some commands are interface specific and may not be available on all platforms listed. Use the online help (**?** in the CLI) to determine if a command is supported by your hardware platform.

In AOS firmware release R11.4.0, support for local firewall processing was implemented. Any time NAT is needed to translate packets typically forwarded by the AOS unit, the full firewall is required (local firewall processing is not sufficient).

In AOS firmware release R11.10.2, the ability to match or block non-initial fragments in ACL entries was implemented.

Configuring IPv4 ACLs and ACPs Using the CLI

The following section gives an overview of the basic steps necessary to create IPv4 ACLs and ACPs through the CLI.

WARNING *Before applying an ACP to an interface, verify your Telnet connection will not be affected by the policy. If a policy is applied to the interface you are connecting through and it does not allow Telnet traffic, your connection will be lost.*

Creating IPv4 ACLs and ACPs to regulate traffic through the routed network requires the following steps. Optional settings are provided at the end of this section for further firewall security, according to your specific needs.

Step 1: Accessing the CLI on page 18

Step 2: Enabling Security Features on page 18

Step 3: Creating an IPv4 ACL and Defining Permissions on page 19

Step 4: Creating and Defining an IPv4 ACP on page 22

Step 5: Applying the IPv4 ACP to an Interface on page 26

Step 6: Enabling the Fast Forwarding Engine on page 27

Step 7: Optional Settings on page 28

Step 1: Accessing the CLI

To access the CLI on your AOS unit, follow these steps:

1. Boot up the unit.
2. Telnet to the unit (**telnet** <*ipv4 address*>). For example:

   telnet 208.61.209.1

NOTE *If during the unit's setup process you have changed the default IPv4 address (**10.10.10.1**), use the configured IPv4 address.*

3. Enter your user name and password at the prompt.

NOTE *The AOS default user name is **admin** and the default password is **password**. If your product no longer has the default user name and password, contact your system administrator for the appropriate user name and password.*

4. Enter the Enable mode by entering **enable** at the prompt as follows:

   >enable
5. Enter your Enable mode password at the prompt.
6. Enter the unit's Global Configuration mode as follows:

```
#config terminal
(config)#
```

Step 2: Enabling Security Features

Use the **ip firewall [vrf** <*name*>**] [local-traffic-only]** command to enable IPv4 AOS security features including ACPs and ACLs, NAT, and the stateful inspection firewall. The optional **vrf** <*name*> parameter enables or disables the firewall for a specific virtual routing and forwarding (VRF) instance. If no VRF is specified, the firewall is enabled on the default VRF. The optional **local-traffic-only** parameter enables the firewall for processing local traffic only. Forwarded traffic is not sent to the firewall when this option is enabled. When the firewall is configured to process local traffic only, NAT is not supported. Use the **no** form of this command to disable the security functionality. The firewall is disabled by default. Enter the command from the Global Configuration mode. For example:

```
(config)#ip firewall
```

The IPv4 firewall is enabled by default in the NetVanta 3100 series.

Step 3: Creating an IPv4 ACL and Defining Permissions

Create an IPv4 ACL and configure it to permit or deny specific traffic. This step determines whether you are creating a standard ACL (matching on source information) or an extended ACL (matching numerous criteria). This step also enters the configuration mode for the IPv4 ACL.

Create an IPv4 ACL

From the Global Configuration mode, create either a standard or extended IPv4 ACL and enter the IPv4 ACL Configuration mode using the following command:

```
(config)#ip access-list [extended | standard] <ipv4 acl name>
```

The *<ipv4 acl name>* parameter names the configured IPv4 ACL using an alphanumeric descriptor that will be referenced within an ACP.

Define Permissions

To configure a **standard** IPv4 ACL, specify the packet source information and decide whether the feature using the IPv4 ACL will apply its action (**permit**) or not (**deny**) to matching traffic using the following command:

```
(config-std-nacl)#[permit | deny] <source> [log] [track <name>]
```

To configure an **extended** IPv4 ACL, specify whether the feature using the ACL will apply its action (**permit**) or not (**deny**) to matching traffic based on protocol, source information, and destination information using the following command:

```
(config-ext-nacl)#[permit | deny] <protocol> <source> <source port> <destination> <destination port>
[log] [track <name>] [fragments]
```

The **permit** parameter indicates traffic matching the criteria is processed. In this case, an **allow**, **discard**, or **nat** entry in the IPv4 ACP will be executed.

The **deny** parameter indicates traffic matching the criteria is not processed any further. In this case the next entry in the IPv4 ACP will be evaluated.

The *<protocol>* parameter specifies the data protocol used by the packet. Valid entries are **ip**, **icmp**, **tcp**, **udp**, **ahp**, **esp**, **gre**, or a specific protocol. Range is **0** to **255**.

The *<source>* parameter specifies the source used for packet matching. Sources can be expressed in one of four ways:

- Using the keyword **any** to match any IPv4 address.
- Using the **host** *<ipv4 ip address>* to specify a single host address. IPv4 addresses should be expressed in dotted decimal notation (for example, **10.10.10.1.**)

- Using the *<ipv4 ip address> <wildcard mask>* format to match all IPv4 addresses in a range. The wildcard mask corresponds to a range of IPv4 addresses (network) or a specific host. Wildcard masks are also expressed in dotted decimal notation (for example, **0.0.0.255**) and they work in reverse logic from subnet masks. When broken out into binary form, a **0** indicates which bits of the IPv4 address to consider, and a **1** indicates which bits are disregarded. For example, specifying 255 in any octet of the wildcard mask equates to a *don't care* for that octet in the IPv4 address.
  Additionally, a 30-bit mask would be represented with the wildcard string **0.0.0.3**, a 28-bit mask with **0.0.0.15**, a 24-bit mask with **0.0.0.255**, and so forth.
- Using the keyword **hostname** *<hostname>* to match traffic based on a domain naming system (DNS) name. The unit must be configured with DNS servers using the Global Configuration mode command **name-server** *<ipv4 address>* for this function to work. Using **vrf** *<name>* in conjunction with the **hostname** parameter associates a nondefault VRF with the DNS host name for the source. The VRF is required if the router's DNS server is on a nondefault VRF. This parameter can only be used with the **hostname** source. The command in this case would appear:

  (config-ext-nacl)#**[permit | deny] hostname** *<hostname>* **[vrf** *<name>***] [track** *<name>***] [log]**

The *<source port>* parameter is optional, and allows you to specify the monitored traffic source port. The source port is used only when the *<protocol>* is specified as **tcp** or **udp**. The following selections are available for specifying source port information:

- Using the keyword **any** matches any port.
- Using the keyword **eq** *<port number/name>* matches only packets equal to a specified source port number.
- Using the keyword **gt** *<port number/name>* matches only packets with a source port number greater than the specified number.
- Using the keyword **lt** *<port number/name>* matches only packets with a source port number less than the specified number.
- Using the keyword **neq** *<port number/name>* matches only packets that are not equal to the specified source port number.
- Using the keyword **range** *<starting port number/name> <ending port number/name>* matches only packets that contain a source port number in the specified range.

  The *<port number/name>* parameter specifies the port number or name used by TCP or UDP to pass information to upper layers. The valid range for port numbers is from **0** to **65535**. All ports below **1024** are considered well-known ports, and are controlled by the Internet Assigned Numbers Authority (IANA). All ports above **1024** are dynamically assigned ports that include registered ports for vendor-specific applications. UDP and TCP ports can also be entered by port name. If the **range** keyword is used, two port values must be entered. The valid entries are listed in *Table 2* for TCP ports and *Table 3* for UDP ports.

Table 2. TCP Port Numbers and Associated Names

bgp (Port 179)	**chargen** (Port 19)	**cmd** (Port 514)
daytime (Port 13)	**discard** (Port 9)	**domain** (Port 53)
echo (Port 7)	**exec** (Port 512)	**finger** (Port 79)
ftp (Port 21)	**ftp-data** (Port 20)	**gopher** (Port 70)
hostname (Port 101)	**https** (Port 443)	**ident** (Port 113)

Table 2. TCP Port Numbers and Associated Names *(Continued)*

irc (Port 194)	**klogin** (Port 543)	**kshell** (Port 544)
login (Port 513)	**lpd** (Port 515)	**nntp** (Port 119)
pim-auto-rp (Port 496)	**pop2** (Port 109)	**pop3** (Port 110)
smtp (Port 25)	**ssh** (Port 22)	**sunrpc** (Port 111)
tacacs (Port 49)	**talk** (Port 517)	**telnet** (Port 23)
time (Port 37)	**uucp** (Port 540)	**whois** (Port 43)
www (Port 80)		

Table 3. UDP Port Numbers and Associated Names

biff (Port 512)	**bootpc** (Port 68)	**bootps** (Port 67)
discard (Port 9)	**dnsix** (Port 195)	**domain** (Port 53)
echo (Port 7)	**isakmp** (Port 500)	**mobile-ip** (Port 434)
nameserver (Port 42)	**netbios-dgm** (Port 138)	**netbios-ns** (Port 137)
netbios-ss (Port 139)	**ntp** (Port 123)	**pim-auto-rp** (Port 496)
rip (Port 520)	**snmp** (Port 161)	**snmptrap** (Port 162)
sunrpc (Port 111)	**syslog** (Port 514)	**tacacs** (Port 49)
talk (Port 517)	**tftp** (Port 69)	**time** (Port 37)
who (Port 513)	**xdmcp** (Port 177)	

The <*destination*> parameter specifies the destination used for packet matching. Destinations can be expressed in the same four ways as the source information (refer to the *<source> parameter on page 19*).

The <*destination port*> parameter is optional, and allows you to specify the monitored traffic destination port. The destination port is used only when the <*protocol*> is specified as **tcp** or **udp**. The same selections available for source port selection are available for destination port selection (refer to the *<source port> parameter on page 20*).

The optional **track** <*name*> parameter associates the IPv4 ACL entry with a particular track. This track can be used to disable the entry in the case of certain events specified by the track.

The optional **log** parameter specifies that any entries that match the IPv4 ACL criteria will be logged.

The optional **fragments** parameter is used to specify that the **ip** protocol ACL entry is only matched by non-initial fragments. The **fragments** keyword is only available when the specified protocol is **ip**. IPv4 ACLs match non-initial fragments in the following manner:

- Non-initial fragments can match entries with the **fragments** keyword, provided the other Layer 3 information specified in the entry matches the packet.
- Non-initial fragments can match entries with the **ip** protocol specified, provided the other Layer 3 information specified in the entry matches the packet.
- Non-initial fragments are implicitly permitted by access groups if the fragments did not match an explicit entry in the ACL.

NOTE *The **fragments** keyword is only useful when the ACL is referenced by an access group. The **fragments** keyword is not needed when the ACL is only used for an ACP. This is because the IPv4 firewall reassembles IPv4 fragment chains prior to inspecting the ACP to determine which action to take on the packets.*

If the **debug access-list** command is issued, a debug message displays (for each entry using the **log** keyword) showing the number of times in the last five seconds that an inspected IPv4 packet has matched that entry. If no IPv4 packet has matched the criteria in the last five seconds, however, no message is displayed. Matches are logged even if they are matched to an entry in the standard ACL cache. Even if the keyword **log** is not specified, a running total of the number of IPv4 packets that have matched the entry's criteria since the entry's original creation can be viewed through the **show ip access-lists** command.

Two caveats to remember:

- Multiple matches for a single packet could be logged if the same IPv4 ACL is used in more than one place in which the IPv4 packet is inspected, such as in both an ACP and a route map.
- If RapidRoute™ is being used (which can only be the case if the IPv4 ACL is being used by an ACP or by an access group), then only the match of the first IPv4 packet in the flow is logged.

Additionally, an ACL comment (using the **remark** <*remark*> command) can be associated with the entry to further describe the purpose of the IPv4 ACL. The descriptive tag allows up to 80 alphanumeric characters enclosed in quotation marks, such as *This list blocks all outbound Web traffic*. The command is entered from the ACL configuration mode.

(config-ext-nacl)#**remark** <*remark*>

You can enter as many criteria as you need to have the IPv4 ACL match traffic for your network. It is important to remember that the order of your entries is crucial: the ACL will match traffic based on criteria from the top down. If the order of the entries is not correct, you will have to remove the applicable IPv4 ACL entries and then reenter them in the correct order. If a new entry needs to be at the top of the entry list, all the previous entries must be removed.

NOTE *If the order of entries in an IPv4 ACL needs to be changed, it is recommended that the GUI be utilized as it offers a way to change the order of entries without removing them all and then adding them back. Refer to Change Order of Entries on page 47 for more information.*

Each IPv4 ACL has an implicit **deny any** as the last criteria if there are other explicit criteria entries within the ACL. Each empty IPv4 ACL has an implicit **permit any** when there are no other explicit criteria entries within the ACL.

Step 4: Creating and Defining an IPv4 ACP

ACPs are used to allow, discard, or manipulate (using NAT) data for each routable IP interface. Each ACP consists of a selector (for example, an ACL) and an action (**allow**, **discard**, or **NAT**). The selector can be an undefined ACL, which will permit any traffic.

To create an IPv4 ACP, enter the following command from the Global Configuration mode and enter the ACP Configuration mode.

(config)#**ip policy-class** *<ipv4 acp name>*

Once the IPv4 ACP is created, configure the action to take when traffic has been received on the interface. Traffic can be permitted to enter the interface by using the **allow list** command or discarded by using the **discard list** command. The destination or source IPv4 address can be translated to a specified IPv4 address by using the **nat source list** or **nat destination list** commands. These commands are covered in detail in the following sections:

- *Allow Traffic Based on IPv4 ACL Entries on page 23*
- *Discard Traffic Based on IPv4 ACL Entries on page 24*
- *Apply NAT to the Destination IPv4 Address on page 25*
- *Apply NAT to the Source IPv4 Address on page 26*

Allow Traffic Based on IPv4 ACL Entries

Use the **allow list** command from the IPv4 ACP Configuration mode to specify an IPv4 ACL to determine which IP packets are allowed to enter the interface to which the IPv4 ACP is assigned, and create a firewall policy session in the IPv4 firewall. All firewall policy sessions are subject to the built-in firewall timers (refer to *Changing Policy Timeouts on page 31*). Additional conditions further define how the traffic is handled and are explained in the following paragraphs.

(config-policy-class)#**allow list** *<ipv4 acl name>* **[self | policy** *<ipv4 acp name>***] [stateless]**

Use the **allow reverse list** command to specify an IPv4 ACL to determine which IP packets are allowed to enter the interface to which the IPv4 ACP is assigned, and create a firewall policy session in the firewall. The **allow reverse list** command is identical in function to the **allow list** command with the exception of the **reverse** keyword. The **reverse** keyword instructs the firewall to use the source information as the destination information and vice versa when attempting matches against the specified IPv4 ACL. This command is most useful when the IPv4 ACP is applied to an interface terminating a VPN tunnel. The **allow reverse list** allows the reuse of the IPv4 ACL defined as the VPN selector.

(config-policy-class)#**allow reverse list** *<ipv4 acl name>* **[self | policy** *<ipv4 acp name>***] [stateless]**

The **self** parameter allows all IP packets passed by the IPv4 ACL and destined for any local interface on the unit to enter the router system. These packets are terminated by the unit and are not routed or forwarded to other destinations. Using the **self** parameter is helpful when opening remote administrative access to the unit (Telnet, secure shell (SSH), ICMP, HTTP, Hypertext Transfer Protocol Secure (HTTPS), etc.).

The **policy** *<ipv4 acp name>* parameter specifies the destination IPv4 ACP against which to match traffic. The firewall attempts to match the specified IPv4 ACP with the IPv4 ACP that is applied to the IP packet's egress interface as determined by the routing table or PBR configuration. This allows configurations to permit packets routed out a single interface, but not the entire system.

The **stateless** parameter is optional and enables bypassing stateful firewall processing and ALGs. It is used for trusted traffic or traffic that the firewall is incorrectly blocking as a perceived attack. Stateless processing is helpful when passing traffic over VPN tunnels. Traffic sent over VPN tunnels is purposely selected and encrypted; there is no need for additional inspection of the traffic by the firewall. VPN configurations created using the VPN Wizard in the GUI use stateless processing by default. If voice quality monitoring (VQM) is being used over VPN, the selectors cannot be stateless.

NOTE *To configure VPN, the firewall must either be disabled or configured to process forwarded traffic. In addition, no VQM statistics are gathered unless the firewall is configured to process all traffic.*

The following example configures the IPv4 ACP named **UNTRUSTED** to allow any traffic that matches the IPv4 ACL named **INWEB** to enter the router system:

```
(config)#ip policy-class UNTRUSTED
(config-policy-class)#allow list INWEB
```

Discard Traffic Based on IPv4 ACL Entries

Use the **discard list** command to specify an IPv4 ACL to determine which IPv4 packets are discarded after being received on the interface to which the IPv4 ACP is assigned. IP packets matched by the IPv4 ACL will be discarded, and no further ACP entries will be inspected. All IP packets not matched by the IPv4 ACL are processed by the next ACP entry or implicitly discarded if no further ACP entries exist.

```
(config)#discard list <ipv4 acl name> [self | policy <ipv4 acp name>]
```

The **self** parameter discards IPv4 packets that are matched by the IPv4 ACL and destined for any local interface on the unit. These packets, had they been allowed, would be terminated by the unit and not routed or forwarded to other destinations. Using the **self** keyword is helpful when forbidding certain access to the unit.

The **policy** *<ipv4 acp name>* parameter specifies the destination IPv4 ACP against which to match traffic. The firewall attempts to match the specified IPv4 ACP with the IPv4 ACP that is applied to the IP packet's egress interface as determined by the routing table or PBR configuration. The **policy** parameter causes all IP packets passed by the IPv4 ACL and destined for the interface using the specified IPv4 ACP to be discarded. If there is no match, the firewall will process the IPv4 packet based on the next ACP entry or implicitly discard it if no further ACP entries exist.

The following example configures the IPv4 ACP named **UNTRUSTED** to discard any traffic that matches the IPv4 ACL named **BLOCK**:

```
(config)#ip policy-class UNTRUSTED
(config-policy-class)#discard list BLOCK
```

NOTE *The IPv4 ACL named **BLOCK** in this example would have to use **permit** statements to specify the matched traffic that will be discarded in the ACP. If **deny** statements were used in the IPv4 ACL, it would cause the next rule in the ACP to be processed instead of dropping the desired traffic.*

Apply NAT to the Destination IPv4 Address

Use the **nat destination list** command to translate the destination IPv4 address to a specified IPv4 address, and create a firewall policy session. The translation is applied only to those IP packets permitted by the specified IPv4 ACL and entering the interface to which the IPv4 ACP is assigned. All firewall policy sessions are subject to the built-in firewall timers (refer to *Changing Policy Timeouts on page 31*).

> (config-policy-class)#**nat destination list** *<ipv4 acl name>* **[address** *<ipv4 ip address>* **| vrf** *<name>* **| port** *<port number>***] [no-alg]**

The **address** *<ipv4 ip address>* parameter specifies the address of the internal IP host to which the translated IPv4 packets are destined. IPv4 addresses should be expressed in dotted decimal notation (for example, **10.10.10.1**).

The optional **port** *<port number>* parameter translates the original destination port to a user-specified port.

The optional **vrf** *<name>* parameter specifies the VRF instance. The VRF does not have to be the same VRF from which the IPv4 packet originated. VRF on an AOS product allows a single physical router to be partitioned into multiple virtual routers. Each router instance has its own route table and interface assignments. Beginning with Release 16.1, all AOS routers supporting multiple VRF instances (multi-VRF) have an unnamed default VRF instance regardless of whether multi-VRF is configured. Therefore, executing the above mentioned commands without specifying a VRF indicates that the specified IPv4 address corresponds to the default unnamed VRF. For more information, refer to the configuration guide *Multi-VRF* (available online at https://supportforums.adtran.com).

The optional **no-alg** parameter allows IP packets matching the IPv4 ACP entry to traverse the firewall without being processed by the ALGs. This parameter, along with the appropriate IPv4 ACL, prevents specific traffic from being processed by the ALGs.

The following example enables NAT for traffic that matches the ACL **INWEB** and changes the destination IPv4 address to **192.168.0.253**:

```
(config)#ip policy-class UNTRUSTED
(config-policy-class)#nat destination list INWEB address 192.168.0.253
```

The **nat destination list** *<ipv4 acl name>* **pool** command is used in conjunction with NAT pools. NAT pools are used to translate IPv4 addresses between a specified internal range and a specified public range for a static one-to-one mapping of addresses. NAT pools is a feature that is outside the scope of this document.

> (config-policy-class)#**nat destination list** *<ipv4 acl name>* **[pool** *<pool name>***] [no-alg]**

NOTE *For more information on NAT pools, refer to the configuration guide NAT Pools in AOS (*available online at https://supportforums.adtran.com*).*

Apply NAT to the Source IPv4 Address

Use the **nat source list** command to translate the source IPv4 address to a user specified IPv4 address (or to the primary IPv4 address of the specified interface) and create a firewall policy session. The NAT is applied only to IP packets permitted by the specified IPv4 ACL, and entering the interface to which the IPv4 ACP is assigned. This function is commonly referred to as a many:1 and is typically used to enable Internet access for hosts on a locally addressed subnet. All firewall policy sessions are subject to the built-in firewall timers (refer to *Changing Policy Timeouts on page 31*).

```
(config-policy-class)#nat source list <ipv4 acl name> [address <ipv4 ip address> | interface
<interface> ] overload [policy <ipv4 acp name>] [no-alg]
```

The **address** *<ipv4 ip address>* parameter specifies the IPv4 address the translated IP packets should be sourced from. IPv4 addresses should be expressed in dotted decimal notation (for example, **10.10.10.1**).

The **overload** parameter allows multiple source IPv4 addresses to be replaced with the single IPv4 address specified or the primary IPv4 address of the specified interface. This conceals internal IPv4 addresses from outside the local network. The **overload** parameter is required when using the **nat source list** command with a single IPv4 address or interface.

The optional **policy** *<ipv4 acp name>* parameter specifies the IPv4 ACP against which to match traffic. The firewall attempts to match the specified IPv4 ACP with the IPv4 ACP that is applied to the IP packet's egress interface as determined by the routing table or PBR configuration. If there is a match, the firewall will process the packet. If there is no match, the firewall will process the packet based on the next ACP entry or implicitly discard it if no further ACP entries exist.

The optional **no-alg** parameter allows IP packets matching the IPv4 ACP entry to traverse the firewall without being processed by the ALGs. This parameter, along with the appropriate IPv4 ACL, prevents specific traffic from being processed by the ALGs.

The **nat source list** *<ipv4 acl name>* **pool** command is used in conjunction with NAT pools. NAT pools are used to translate IPv4 addresses between a specified internal range and a specified public range for a static one-to-one mapping of addresses. NAT pools is a feature that is outside the scope of this document.

```
(config-policy-class)#nat source list <ipv4 acl name> pool <pool name>
```

NOTE *For more information on NAT pools, refer to the configuration guide NAT Pools in AOS (*available online at https://supportforums.adtran.com*).*

Step 5: Applying the IPv4 ACP to an Interface

The ACP has no affect until it is activated by applying it to an interface and enabling the firewall. To apply the IPv4 ACP to an interface, navigate to the desired interface configuration mode and enter the **ip access-policy** *<ipv4 acp name>* command. The following example assigns the ACP **UNTRUSTED** to the Ethernet 0/1 interface:

```
(config)#interface ethernet 0/1
(config-eth 0/1)#ip access-policy UNTRUSTED
```

Step 6: Enabling the Fast Forwarding Engine

RapidRoute is ADTRAN's fast forwarding engine. It is a packet processing architecture in routers that classifies packets into packet flows based upon the protocol used by the packet, the source and destination IPv4 address, and the protocol-specific information, such as source and destination port numbers. Packet flows are defined as the unidirectional representation of a conversation between two IP hosts, and each ingress interface maintains a traffic flow table. The identifiers in the flow tables are the same as those in the firewall association table, which allows one-to-one mapping between a flow entry and the firewall's association selector. Using RapidRoute allows the router to process traffic more quickly, because as each packet is classified, it is placed in a traffic flow of other packets with similar features. This means each packet is classified only once, rather than classified every time it is used by an AOS feature, such as the firewall, VPN, NAT, etc. RapidRoute is a beneficial routing enhancement, especially in instances where traffic must be prioritized, delivered on quality of service (QoS) requirements, or kept from monopolizing bandwidth. Using RapidRoute especially in conjunction with the AOS firewall can greatly improve performance. Refer to *Table 4 on page 27* for a complete list of available RapidRoute options.

To enable RapidRoute on an interface, use the **ip ffe** command from the interface configuration mode prompt. This command should be applied to all active IP interfaces. For example:

```
(config)#interface ethernet 0/1
(config-inf-eth 0/1)#ip ffe
(config-inf-eth 0/1)#interface ppp 1
(config-inf-ppp 1)#ip ffe
```

Table 4. FFE Commands

Command	Explanation
ip ffe max-entries *<value>*	The **ip ffe max-entries** command is used to set the global maximum number of RapidRoute entries allowed. Use the **no** form of this command to return to the default value.
ip ffe timeout tcp *<max timeout> <inactive timeout>*	The **ip ffe timeout tcp** command specifies the timeout values in seconds for TCP. Use the **no** form of this command to return to the default value.*
ip ffe timeout udp *<max timeout> <inactive timeout>*	The **ip ffe timeout udp** command specifies the timeout values in seconds for UDP. Use the **no** form of this command to return to the default value.*
ip ffe timeout icmp *<max timeout> <inactive timeout>*	The **ip ffe timeout icmp** command specifies the timeout values in seconds for ICMP. Use the **no** form of this command to return to the default value.*
ip ffe timeout ah *<max timeout> <inactive timeout>*	The **ip ffe timeout ah** command specifies the timeout values in seconds for AH protocol. Use the **no** form of this command to return to the default value.*
ip ffe timeout esp *<max timeout> <inactive timeout>*	The **ip ffe timeout esp** command specifies the timeout values in seconds for ESP protocol. Use the **no** form of this command to return to the default value.*
ip ffe timeout gre *<max timeout> <inactive timeout>*	The **ip ffe timeout gre** command specifies the timeout values in seconds for GRE protocol. Use the **no** form of this command to return to the default value.*
ip ffe timeout other *<max timeout> <inactive timeout>*	The **ip ffe timeout other** command specifies the timeout values in seconds for protocols not listed. Use the **no** form of this command to return to the default value.*

Table 4. FFE Commands *(Continued)*

Command	Explanation
* *The <max timeout> variable specifies maximum timeout in seconds. This is the maximum amount of time an entry will be kept in the RapidRoute table regardless of activity. Valid range is **60** to **86400** seconds. The <inactive timeout> variable specifies idle timeout in seconds. This is the amount of time an entry will remain in the RapidRoute table with no additional activity. Valid range is **10** to **86400** seconds.*	

Step 7: Optional Settings

There are optional configuration settings in addition to the previous steps that require further explanation. These settings are not necessary for all network environments and are dependent upon the preferred security methods in use. This section expands on the following additional IPv4 firewall configurations:

- *Enabling and Disabling Attack Checking on page 28*
- *Enabling and Disabling Application-Level Gateways on page 29*
- *Enabling Stealth Mode on page 31*
- *Changing Policy Timeouts on page 31*
- *Changing the Number of Policy Sessions Allowed on page 32*
- *Enabling Fast NAT Failover on page 33*

Enabling and Disabling Attack Checking

As described in *Attack Checking on page 2*, attack checking can be enabled or disabled for IP spoofing, reflexive traffic, reverse path forwarding (RPF), SYN flooding, TCP RST sequence number checking, and WINNUKE. Configuration information is provided below for each of these attack checks.

Reflexive traffic refers to IP packets that are routed out of the same interface on which they arrived. The default AOS behavior in this case is to let the packet bypass the firewall and to send it out without creating a firewall association for it. If reflexive traffic checking is enabled, however, the packet will be processed normally by the firewall. Use the **no** form of this command to disable the attack check. To enable IPv4 reflexive traffic checking, use the **ip firewall check reflexive-traffic** command from the Global Configuration mode prompt. For example:

(config)#**ip firewall check reflexive-traffic**

RPF is a spoofing check that uses a route lookup to verify that IP traffic has entered on the appropriate interface. By default, RPF checking is enabled. RPF checking should be disabled if your application allows IP traffic to arrive on an interface sourced from networks contradicting the route table. This feature can be disabled on a per ACP basis by issuing the **no** form of the command in conjunction with the IPv4 ACP name you do not want to be checked. For example:

(config)#**no ip policy-class** *<ipv4 acp name>* **rpf-check**

Enable RPF checking with the following command:

(config)#**ip policy-class** *<ipv4 acp name>* **rpf-check**

SYN flooding is a well-known denial of service (DoS) attack on TCP-based services. TCP requires a three-way handshake before actual communication begins between two hosts. A server must allocate resources to process new connection requests that are received. A potential attacker is capable of transmitting large numbers of service requests in a short period of time without completing the three-way handshake in a timely manner. The servers allocate all resources to process the incomplete requests. When SYN flooding attack checking is enabled, the firewall is allowed to limit TCP service requests and allow only legitimate requests to pass through. Use the **no** form of this command to disable the attack check. By default, SYN flooding check is enabled. If the SYN flooding check has been disabled, it can be enabled for IPv4 by issuing the **ip firewall check syn-flood** command from the Global Configuration mode. For example:

```
(config)#ip firewall check syn-flood
```

TCP sequence number checking prevents a TCP reset packet with a sequence number outside the TCP receive window from being processed by the firewall. The sequence number is learned through stateful packet inspection of the TCP traffic. By default, TCP reset sequence number checking is enabled. To enable for IPv4, use the **ip firewall check rst-seq** command from the Global Configuration mode prompt. Use the **no** form of this command to disable the attack check. For example:

```
(config)#ip firewall check rst-seq
```

The WinNuke attack is a well-known DoS attack on hosts running older versions of the Microsoft Windows® operating systems, such as Windows 95, Windows NT, and Windows 3.1x. An intruder would send out-of-band (OOB) data over an established connection to a Windows user. This data is denoted by setting the URG (urgent) flag in the TCP header. Older versions of Windows cannot properly handle the OOB data, causing the host under attack to react unpredictably. Normal shut down of the hosts would generally return all functionality. Once enabled, the firewall is configured to discard all OOB data to prevent network problems. By default, WinNuke attack checking is disabled. To enable, use the **ip firewall check winnuke** command from the Global Configuration mode prompt. Use the **no** form of this command to disable the attack check. For example:

```
(config)#ip firewall check winnuke
```

Enabling and Disabling Application-Level Gateways

Certain ALGs can be enabled or disabled, depending on your specific network requirements. By default, the ALG for FTP, IRC, PPTP, and SIP are enabled. Conversely, the ALG for MSN, MSZONE, and H.323 are disabled by default. The SIP ALG is only present on ADTRAN router and switch products, not voice products.

NOTE *Disabling an ALG could cause the firewall to block some of the traffic for the specified protocol.*

To enable the ALGs, use the **ip firewall alg** command, followed by the specific ALG, as shown below:

```
(config)#ip firewall alg ftp
(config)#ip firewall alg h323
(config)#ip firewall alg irc
(config)#ip firewall alg msn
(config)#ip firewall alg mszone
(config)#ip firewall alg pptp
(config)#ip firewall alg sip
```

SIP ALG

The SIP ALG performs deep packet inspection on SIP and Session Description Protocol (SDP) payloads of SIP requests destined for UDP port 5060 and SIP responses to that traffic. If necessary, the ALG translates IPv4 addresses and ports in SIP and SDP fields using source or destination NAT. The ALG uses information it has read from SIP and SDP fields to initiate appropriate policy sessions to allow applicable SIP responses, RTP, and Real-Time Transport Control Protocol (RTCP) to traverse the firewall.

NOTE *The SIP ALG is only available on AOS data products (not AOS voice products).*

NOTE *The SIP ALG cannot be used if the initial SIP traffic is not destined for UDP port 5060.*

Certain and recent updates to RFC 3261 might not be supported through the SIP ALG. If an application does not work correctly with the SIP ALG, you can try disabling the SIP ALG by issuing the **no ip firewall alg sip** command from the Global Configuration mode and retry the application. If the application still does not function, then try enabling SIP proxy. Refer to the configuration guide *Configuring the SIP Proxy in AOS* (available online at https://supportforums.adtran.com) for more information on SIP Proxy.

HTTP URL Filter ALG

The HTTP URL filter ALG can be used to block or allow certain URLs when accessed by certain hosts. In order for this ALG to function, it must (after it has been properly configured) be applied to an interface's inbound or outbound direction.

The HTTP URL filter ALG checks are made in the following order:

1. Check to see if the URL domain is configured as an exclusive domain. If so, either **permit** or **deny** the request as specified. If not, go to Step 2.
2. If Websense server is configured and reachable, send the URL request to the Websense server and either **permit** or **deny** the request as specified by the response from Websense. If not, go to Step 3.
3. If **allowmode** is enabled, **permit** the request. If it is disabled, block the request.

This ALG implements a partial TCP stack to send required TCP messages when a URL request is denied. When a deny occurs, a standard block page is sent to the client stating that the request was denied. Also, a TCP RST message is sent to both the client and the Web server so that they close the connection between them (if such a connection exists). Refer to *URL Filtering/Top Websites Reporting* (available online at https://supportforums.adtran.com) for more information on URL filtering.

Enabling Stealth Mode

In order to determine if there are open ports on the unit that will allow traffic from an outside source, an attacker will systematically send either TCP or UDP traffic to various ports and look at the responses. Under normal circumstances, the unit sends back a **TCP RST** (if TCP traffic has been denied) and **ICMP port unreachable** (if UDP traffic has been denied). These messages help an attacker narrow down the list of possible vulnerable ports. In AOS, this can be avoided by placing the unit in stealth mode, in which case no messages will be sent back to the source if traffic is denied. It should be noted that the unit cannot detect the presence of a port scan. It can only prevent any port scans that are executed from being successful. It should also be noted that the stealth mode does not apply to static filters (access groups and ACPs).

To enable stealth mode, use the **ip firewall stealth** command from the Global Configuration mode prompt. For example:

```
(config)#ip firewall stealth
```

Changing Policy Timeouts

Session filtering based on inactivity can sometimes occur sooner than is desirable. Use the **ip policy-timeout** command to customize timeout intervals for TCP, UDP, ICMP, AH, ESP, and GRE protocols or specific services (by listing the particular port number).

The example below creates customized policy timeouts for the following services:

- WWW (TCP port 80): timeout 24 hours (86400 seconds)
- Telnet (TCP port 23): timeout 20 minutes (1200 seconds)
- FTP (TCP port 21): timeout 5 minutes (300 seconds)
- All other TCP services: timeout 8 minutes (480 seconds)

```
(config)#ip policy-timeout tcp www 86400
(config)#ip policy-timeout tcp telnet 1200
(config)#ip policy-timeout tcp ftp 300
(config)#ip policy-timeout tcp all-ports 480
```

Policy timeouts for post-connection TCP policy sessions can also be configured. A TCP policy session enters a post-connection state after the receipt of a bidirectional FIN or after the receipt of an RST.

Use the **ip firewall fin-timeout** command to configure the timeout for a policy session closed by a bidirectional TCP FIN.

```
(config)#ip firewall fin-timeout <value>
```

Use the **ip firewall rst-timeout** command to configure the policy timeout for a policy session closed by a TCP RST.

```
(config) #ip firewall rst-timeout <value>
```

The <*value*> parameter specifies the time period in seconds allowed for TCP FIN or RST. Range is **0** to **4294967295** seconds. A value of zero indicated that the policy session should be deleted immediately without entering a post-connection state. This could be necessary for hosts that do not implement the TIME_WAIT TCP state correctly, but instead permit immediately reopening closed sessions.

Changing the Number of Policy Sessions Allowed

A global value for the maximum number of policy sessions (including both pending and non-pending) allowed in the AOS device at one time is calculated at boot time based on amount of RAM in the AOS device. The maximum number of sessions allowed per ACP (where a session is associated with the ingress ACP for the initial traffic that created the session) defaults to 45 percent of this global value.

NOTE *The commands **show ip policy-stats** and **show ip policy-class** display the number of policy sessions currently on the unit, as well as the calculated maximums (both global and per ACP). See Table 7 on page 76 for more information on these commands.*

There are several ways in which users can override these maximum policy session values:

Use the **policy-class max-sessions** command to specify the maximum number of allowed policy sessions in the AOS product for both IPv4 and IPv6 combined. This command sets the maximum session limit for ALL ACPs on the AOS unit.

```
(config)#policy-class max-sessions <number>
```

The <*number*> parameter specifies the number of allowed ACP sessions. Valid range is 1 to a value based on the amount of RAM in the AOS device (see *Table 5 on page 32*).

Table 5. Default Values Based on RAM

RAM Amount	Default Max Sessions
64 MB	10000
128 MB	30000
256 MB	80000
512 MB	200000
768 MB	300000
1 GB	450000

Use the **ip policy-class** <*ipv4 acp name*> **max-sessions** command to specify the maximum number of allowed policy sessions for the named IPv4 ACP.

```
(config)#ip policy-class <ipv4 acp name> max-sessions <number>
```

The <*ipv4 acp name*> parameter identifies the configured IPv4 ACP using an alphanumeric descriptor (maximum of 50 characters). All ACP descriptors are case sensitive.

The <*number*> parameter specifies the maximum number of allowed policy sessions for the named IPv4 ACP. This number must be within the appropriate range limits. The limits depend on the type of AOS device being used. Setting this value to **0** restores the default setting. The maximum number of sessions available for all IP versions (IPv4 and IPv6) and all ACPs is determined at boot time based on the amount of RAM in the AOS device. For a named IPv4 ACP, this default is 45 percent of the total number of allowed ACP sessions.

NOTE *The per ACP maximum can be overridden by either a lower value or by a higher value that is less than or equal to the global maximum. The global maximum can be overridden by a lower value than calculated, but not by a higher value.*

Use the **ip policy-class max-host-sessions** command to specify the maximum number of allowed IPv4 ACP sessions that can be created from each unique source address. This command is used in conjunction with a named IPv4 ACP and only applies the limit to that particular ACP.

(config)#**ip policy-class** <*ipv4 acp name*> **max-host-sessions** <*number*>

The <*ipv4 acp name*> parameter identifies the configured IPv4 ACP using an alphanumeric descriptor (maximum of 50 characters). All ACP descriptors are case sensitive.

The <*number*> parameter specifies the maximum number of allowed ACP sessions that can be created from each unique source address. The number must be within the appropriate range limits. The limits depend on the type of AOS device being used. Setting this value to 0 restores the default setting. By default, this feature is turned off (meaning no limits per source address will be enforced).

NOTE *Ensuring that no source host is allowed to create an inordinate number of sessions can help guard against malicious attacks.*

Enabling Fast NAT Failover

Use the **ip firewall fast-nat-failover** command to enable the IPv4 firewall to perform failover on a policy-session so that incorrect associations are not used. Fast NAT failover only applies to source NAT (rather than destination NAT), and it applies to situations in which the IPv4 address of the destination interface (the selector 2 interface on the public side of the unit) changes or the selector 2 ACP changes completely (refer to *Active Policy Sessions on page 5* for more information on selectors). For example, fast NAT failover would be used when the current destination interface receives a new IPv4 address or when the interface changes to an interface with a different applied IPv4 ACP. In these situations, the policy session needs to be deleted quickly so that traffic will not continue to use association information that is no longer accurate. Fast deletion of the policy session also ensures that incorrect source NAT information is not advertised by the AOS device.

When a change to the destination interface's IPv4 address occurs and the next IPv4 packet of the session arrives, a new policy session with the updated NAT information is created. The association change is not technically considered a rework since the old policy session is torn down and the new policy session is brought up in its place. However, the change is similar to a rework because of how quickly the new session replaces the old session.

Fast NAT failover is turned off by default to allow policy sessions to be reused when necessary. This is useful in the case of interface flapping (when an interface goes down and then comes back up) because the destination information that should be used will be exactly the same when the interface is operational again. It would be a waste of resources to tear down the session and then immediately bring up a session with the exact same characteristics. For the same reason, leaving fast NAT failover off is useful if the destination interface must occasionally refresh its DHCP address and the same IPv4 address is always expected to be received from the DHCP server. Since fast NAT failover has no way of knowing that the new IPv4 address is the same as the old one, the result would be an unnecessary failover action on the old policy session.

There are several common applications in which fast NAT failover is useful. In the case of load sharing, there is one specific load shared interface that is the *sticky interface* for the selector, or the interface with which the selector is associated. If this interface is deleted, then the IPv4 firewall will need to perform failover on the policy session so load shared traffic will not continue to use that interface. Another similar application is when the AOS device is using dial backup (meaning AOS has a backup interface for another interface). When the destination interface is changed, the IPv4 firewall needs to delete the policy sessions that send traffic out the old interface. Additionally, when the AOS device is using DHCP, the destination interface is not expected to always be given the same address and the IPv4 firewall must make sure that failover is performed so that incorrect associations are not used.

Enable fast NAT failover using the following command:

```
(config)#ip firewall fast-nat-failover
```

Firewall Order of Operations

This section describes the various segments of the data path that packets can take when being processed and routed in AOS. Different processing paths can be taken depending on variables such as whether the firewall or ALGs are enabled, if the packet is a VPN packet, if the packet is a TCP packet, etc.

When a packet is processed by the firewall, it can be sent through the *slow path* or the *fast path*. Packets processed in the fast path are scrutinized less than those processed in the slow path. As a result, the fast path throughput is faster than the slow path. The first packet in a traffic flow is sent through the slow path. After that, a packet will only be sent through the fast path if no ALGs are being used, if the packet is not destined for self, and, in the case of TCP traffic, if the packet is part of an established session. See *Order of Processing in the IPv4 Firewall on page 35* for a detailed description of the checks used in the slow path.

The IPv4 firewall contains a reassembly engine. This allows the firewall to inspect the full contents of fragmented packets that traverse the router. The reassembly engine is also used to reassemble fragments received at the router’s local interfaces, even if the firewall is disabled. IP fragments are pieces of an IP packet that are too large to transmit all at one time. Reassembly takes place when the AOS device receives or forwards two or more IP fragments. If the fragments are contained within VPN packets, the packets must be decrypted before the fragments can be reassembled (in the same way, if the AOS device sends out a large VPN packet, it must first fragment the packet and then encrypt the fragments).

RapidRoute

RapidRoute is a processing option that is available outside of the firewall. Packets that go through a RapidRoute entry do not pass through the firewall, meaning that they use neither the fast path nor the slow path. Use the **ip ffe** command to enable RapidRoute on an Ethernet interface.

```
(config-eth 0/1)#ip ffe
```

Use the **no** form of this command to disable this feature.

```
(config-eth 0/1)#no ip ffe
```

A traffic flow can only create a RapidRoute entry if an active policy session for that traffic flow has been created. The first packet in a traffic flow always passes through the firewall. Once a policy session exists, then the first matching packet that arrives from either direction will cause a RapidRoute entry to be created for that direction. A RapidRoute entry is only created if the packet would have normally been processed through the fast path of the firewall. Once a RapidRoute entry is created, any matching packets arriving on the interface are not processed by the firewall, but are sent through the RapdiRoute entry and routed out of the AOS device quickly. Exceptions to this rule are discussed in the following paragraphs. There can be up to two RapidRoute entries corresponding to an active policy session, one for each direction a packet could be traveling.

If the IPv4 firewall determines that the packet needs to be processed through the slow path, then RapidRoute will not be used and the packet will be sent through the firewall. In addition, there are certain other conditions where a packet matches the RapidRoute entry, but will not be processed by it. This will be the case if the packet is an IPv4 fragment, since it may have to be collected along with other fragments in the firewall and reassembled. This will also be the case if the packet header specifies any IP options, since the option might require inspection by the firewall. If the packet has an incorrect header length, a mismatch between packet length and the IPv4 header packet length, or a TTL less-than-or-equal-to zero, then the packet will simply be dropped.

The default inactivity timeout for a RapidRoute entry, regardless of the protocol of the traffic, is 15 seconds, which is less than the default timeout for policy sessions. Unlike firewall associations, a RapidRoute entry takes into account the type of service (TOS) field in the packet header when deciding how to route a packet. This only matters if PBR that matches the TOS field is implemented.

NOTE *RapidRoute stays in sync with PBR.*

Order of Processing in the IPv4 Firewall

The series of criteria where the IPv4 firewall exhaustively scrutinizes IP packets in a traffic flow is called the *slow path*. The following list details the process order and criteria for the *slow path* through the IPv4 firewall:

1. If a VPN packet is destined for the AOS device with a crypto map on the interface and a matching security agreement (SA) is found, then decrypt the packet.
2. Perform simple attack checks for invalid IP options, invalid lengths (including transport protocol headers), null TCP packets, etc.
3. Verify if a packet matches an existing policy session. The active policy sessions on the source VRF/ACP should be checked first, followed by the pending policy sessions. The packets are matched based on protocol, the source/destination IPv4 addresses, and the source/destination ports. If a packet does not match an existing policy session, then a new active policy session is created.
4. Perform TCP attack checks, looking for conditions such as an unexpected sequence number or a post-connection SYN.

5. Handle TCP retransmits.
6. Process ALGs.
7. Process transport protocol specifics:
   - TCP - sequence number checks, maximum segment size (MSS) modifications, and checks for flags (FIN, RST, etc.)
   - UDP - no action needed
8. Make necessary adjustments to the transport protocol checksum.
9. NAT is applied, if required.
10. If VPN outbound processing is needed, encrypt the packet.

Route Lookups

Although routing itself is performed outside the firewall, the AOS device must perform route lookups at certain key points when processing a packet in the firewall (for example, the device must locate the outgoing interface for the packet). Route lookups are especially important for the implementation of NAT and PBR.

The ingress ACP's selector contains the destination interface(s) (See *Figure 1 on page 5* to gain a better understanding of the selector interface). When the AOS device learns the destination interface for an IP packet that is in the IPv4 firewall, it also learns the packet's destination ACP. For example, when processing an ICMP packet on which the firewall is performing a normal allow or a source NAT, the AOS device uses the route lookup to see if a route to the packet's destination IPv4 address can be determined. If a route is not found, then the destination ACP is invalid and the AOS device will drop the packet.

When the IPv4 firewall performs a normal destination NAT or any type of NAT where the VRFs are being changed, the AOS device carries out a route lookup using the new NAT/VRF information. This lookup allows the policy session's selectors to be updated appropriately.

When performing PBR, the IPv4 firewall checks to see if the packet matches an applicable route map. If so, then the AOS device can change the packet's destination interface (if the route map has been configured to do so). If NAT is also being performed, the AOS device needs to make sure that PBR is making decisions based on the packet's internal address. The internal address for source NAT is the pre-NAT address and the internal address for destination NAT is the post-NAT address.

NOTE

When load sharing is enabled with the IPv4 firewall, AOS must act in a certain manner. For more information refer to the quick configuration guide Configuring IP Load Sharing in AOS (available online at https://supportforums.adtran.com).

Configuring the IPv4 Firewall Using the GUI

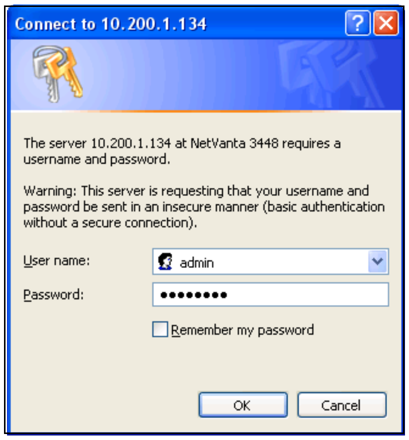
The GUI AOS IPv4 firewall menus allow an administrator to quickly configure an initial firewall policy, change default protocol and traffic timeouts, and configure advanced ACPs to control traffic going through the IPv4 firewall.

Accessing the GUI

1. To begin configuring the IPv4 firewall using the GUI, connect to the GUI following these steps:
2. Open a new Web page in your Internet browser.
3. Type your AOS product’s IP address in the Internet browser’s address field in the following form: **http://**<*ip address*>. For example:

   http://10.200.1.134

4. At the prompt, enter your user name and password and select OK.



Connect to 10.200.1.134

The server 10.200.1.134 at NetVanta 3448 requires a username and password.

Warning: This server is requesting that your username and password be sent in an insecure manner (basic authentication without a secure connection).

User name: admin

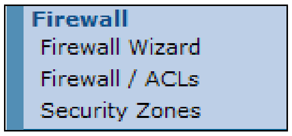
Password: ••••••••

Remember my password

*The default user name is **admin** and the default password is **password**.*

Firewall Menus

The AOS GUI breaks down the IPv4 firewall configuration into three separate menus: Firewall Wizard, Firewall/ACLs, and Security Zones. Each menu focuses on a different aspect of the IPv4 firewall.



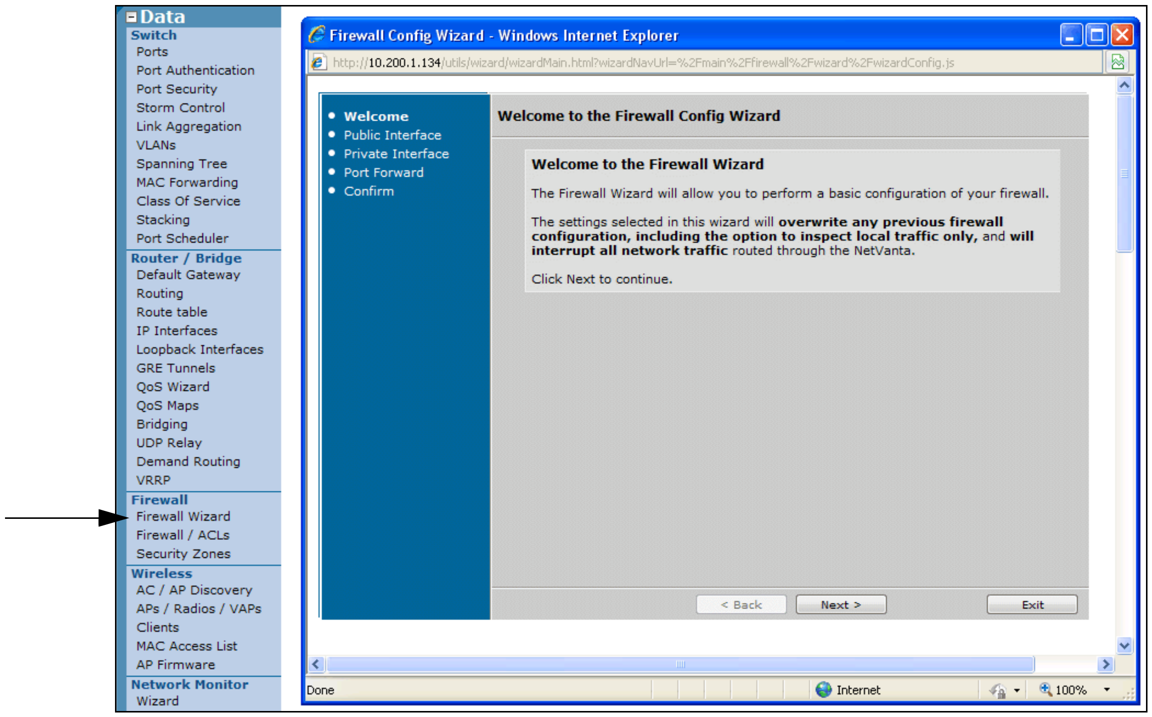
Firewall Wizard

The Firewall Wizard can be used for the initial IPv4 firewall policy configuration. An administrator can quickly enable Internet sharing (using NAT) by selecting the public interface and then selecting the internal interfaces that will use the public interface for outbound traffic. Port forwarding can also be configured in the wizard if there are servers (Web, e-mail, etc.) on the internal network that need to be accessed from the Internet.

WARNING	*The **Firewall Wizard** should only be used for the initial firewall configuration. The wizard will overwrite any existing firewall policies and temporarily interrupt all network traffic. The **Security Zones** firewall menu in the GUI can be used to modify existing IPv4 firewall policies.*

To configure the IPv4 firewall using the Firewall Wizard, follow these steps:

1. Navigate to **Data** > **Firewall Wizard** in the menu on the left of the GUI. A new window with the Firewall Wizard welcome menu will appear. Select **Next** to confirm that you understand that the settings selected using the wizard will overwrite any previous firewall configuration, including any local firewall configuration.



Welcome to the Firewall Config Wizard

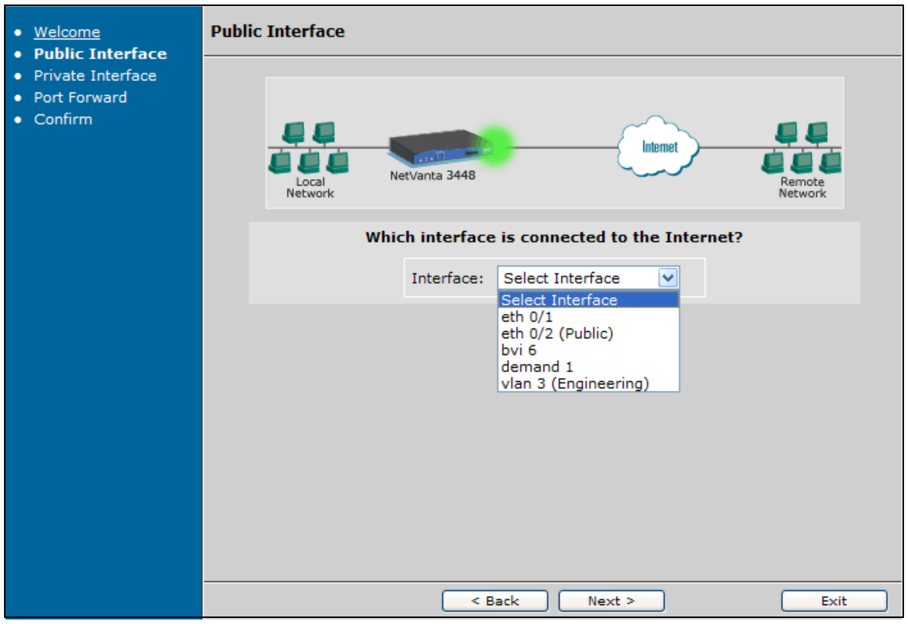
Welcome to the Firewall Wizard

The Firewall Wizard will allow you to perform a basic configuration of your firewall.

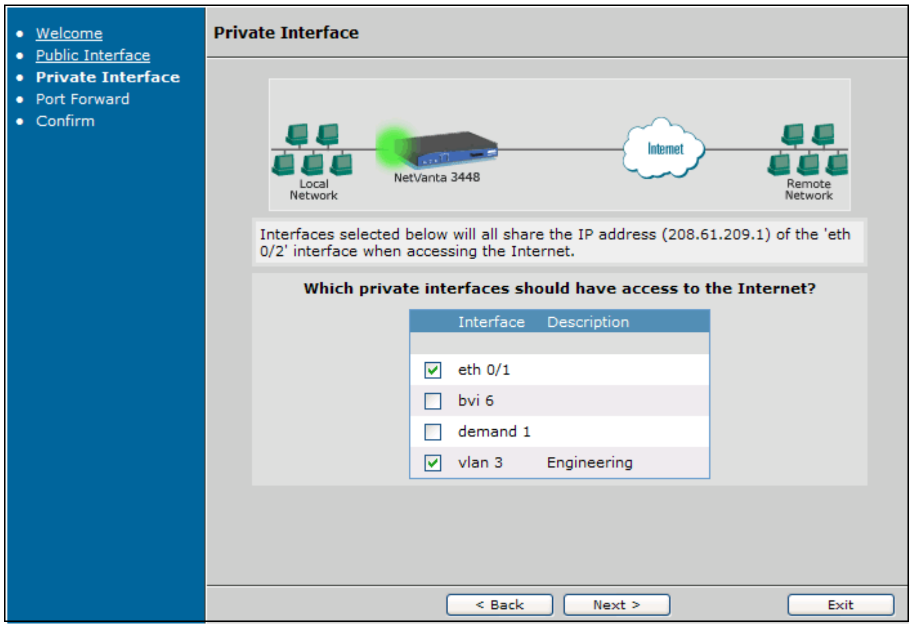
The settings selected in this wizard will **overwrite any previous firewall configuration, including the option to inspect local traffic only,** and **will interrupt all network traffic** routed through the NetVanta.

Click Next to continue.

2. In the **Interface** drop-down menu, select the interface on the unit that is connected to the Internet. Then select **Next**.



3. Select all of the private (internal) interfaces that will use the public interface to get out to the Internet by checking the associated check box. Then select **Next**.

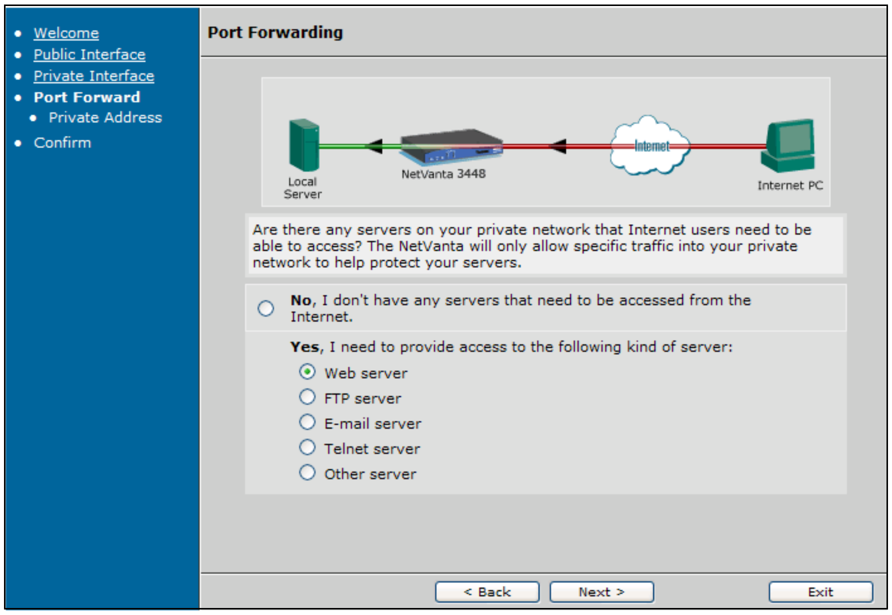


Interfaces selected below will all share the IP address (208.61.209.1) of the 'eth 0/2' interface when accessing the Internet.

Which private interfaces should have access to the Internet?

	Interface	Description
✓	eth 0/1	
☐	bvi 6	
☐	demand 1	
✓	vlan 3	Engineering

4. Select the radio button next to the server type (if any) on your private network that you want to allow access to from the Internet. Then select **Next**. Only one server can be selected at a time. If there are no servers on the private network that need to be accessed from the Internet, then proceed to *Step 6 on page 41*.



Are there any servers on your private network that Internet users need to be able to access? The NetVanta will only allow specific traffic into your private network to help protect your servers.

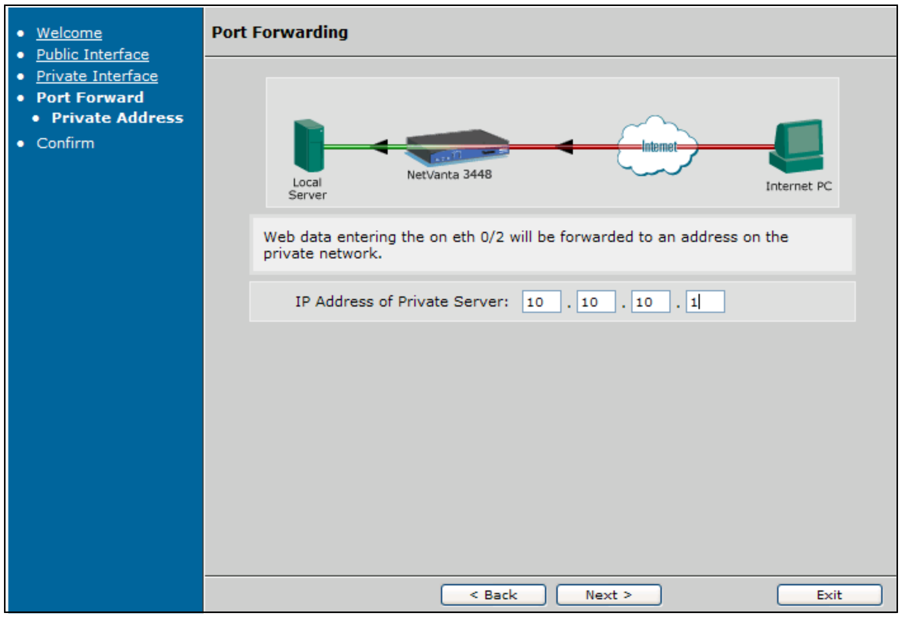
No, I don't have any servers that need to be accessed from the Internet.

Yes, I need to provide access to the following kind of server:

[✓ Web server]
FTP server
E-mail server
Telnet server
Other server

5. Specify the private IPv4 address of the server that will receive the forwarded traffic. Select **Next** and the screen will return to the server selection screen as shown in *Step 4 on page 40*.

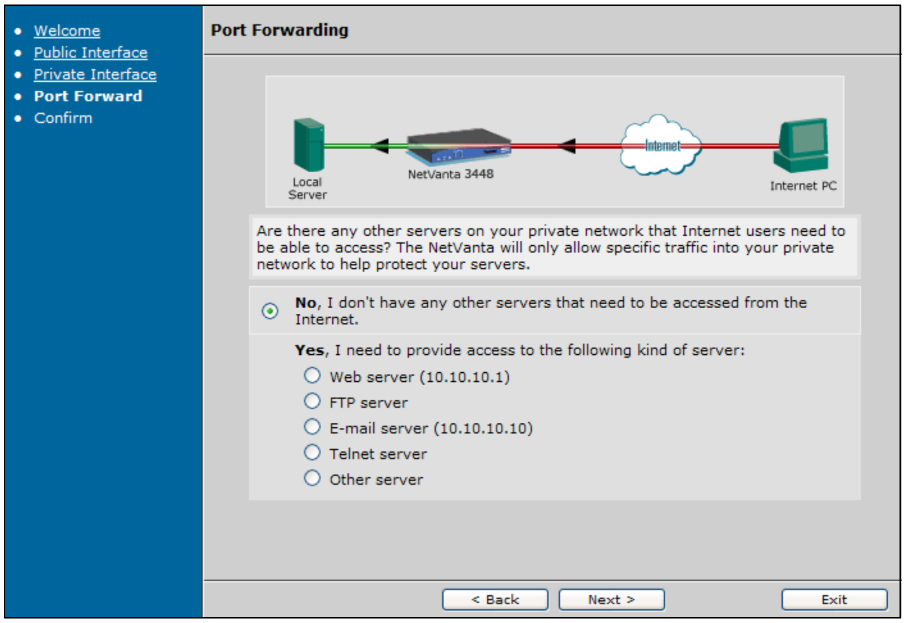
NOTE *Identify any other servers that will be located on the private network (if any). Repeat Step 4 and 5 until all servers on the private network that will be accessed from the Internet have been entered.*



Web data entering the on eth 0/2 will be forwarded to an address on the private network.

IP Address of Private Server: 10 . 10 . 10 . 1

6. Select **No** to indicate that there are no servers that need to be accessed from the Internet or after all servers that need to be accessed from the Internet have been entered. Select **Next**.



Port Forwarding

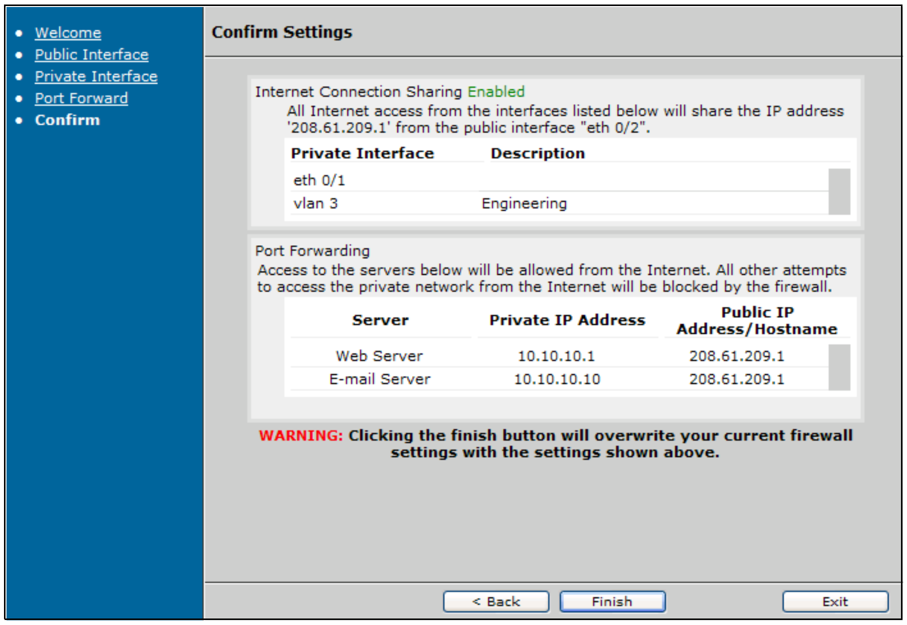
Are there any other servers on your private network that Internet users need to be able to access? The NetVanta will only allow specific traffic into your private network to help protect your servers.

[✓ **No**, I don't have any other servers that need to be accessed from the Internet.]

Yes, I need to provide access to the following kind of server:

- Web server (10.10.10.1)
- FTP server
- E-mail server (10.10.10.10)
- Telnet server
- Other server

7. Confirm the new firewall settings. Select **Finish** to overwrite existing firewall policies with the new firewall settings.



Confirm Settings

Internet Connection Sharing Enabled

All Internet access from the interfaces listed below will share the IP address '208.61.209.1' from the public interface "eth 0/2".

Private Interface	Description
eth 0/1	
vlan 3	Engineering

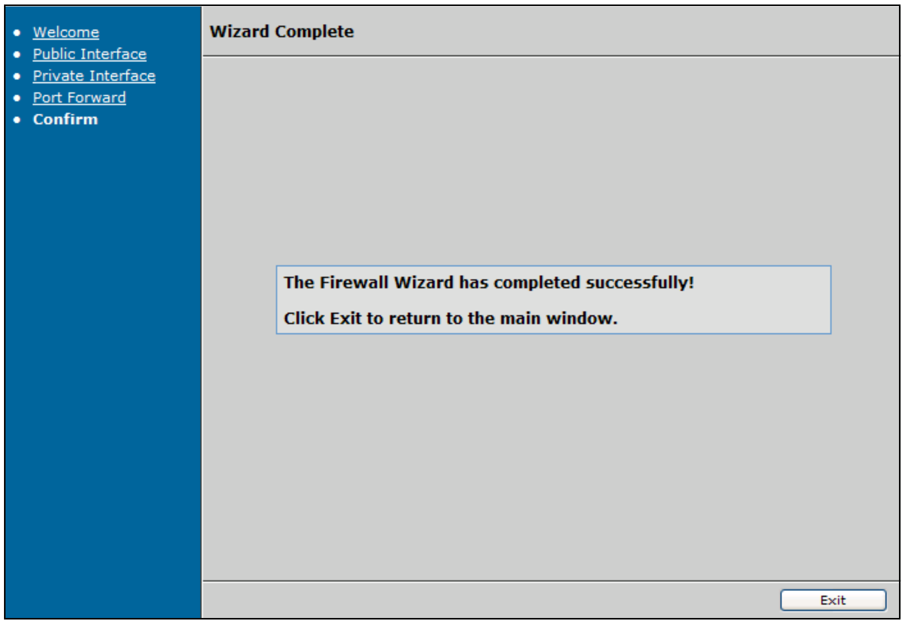
Port Forwarding

Access to the servers below will be allowed from the Internet. All other attempts to access the private network from the Internet will be blocked by the firewall.

Server	Private IP Address	Public IP Address/Hostname
Web Server	10.10.10.1	208.61.209.1
E-mail Server	10.10.10.10	208.61.209.1

WARNING: Clicking the finish button will overwrite your current firewall settings with the settings shown above.

8. A final message indicates that the Firewall Wizard has completed successfully. Select **Exit** to return to the main menu.



The Firewall Wizard has completed successfully!

Click Exit to return to the main window.

NOTE *Remember to save the configuration to ensure the settings will not be lost after a restart. To save the configuration, close the Firewall Wizard by selecting* ***Exit*** *and then select* ***Save*** *in the upper right-hand corner of the GUI heading.*

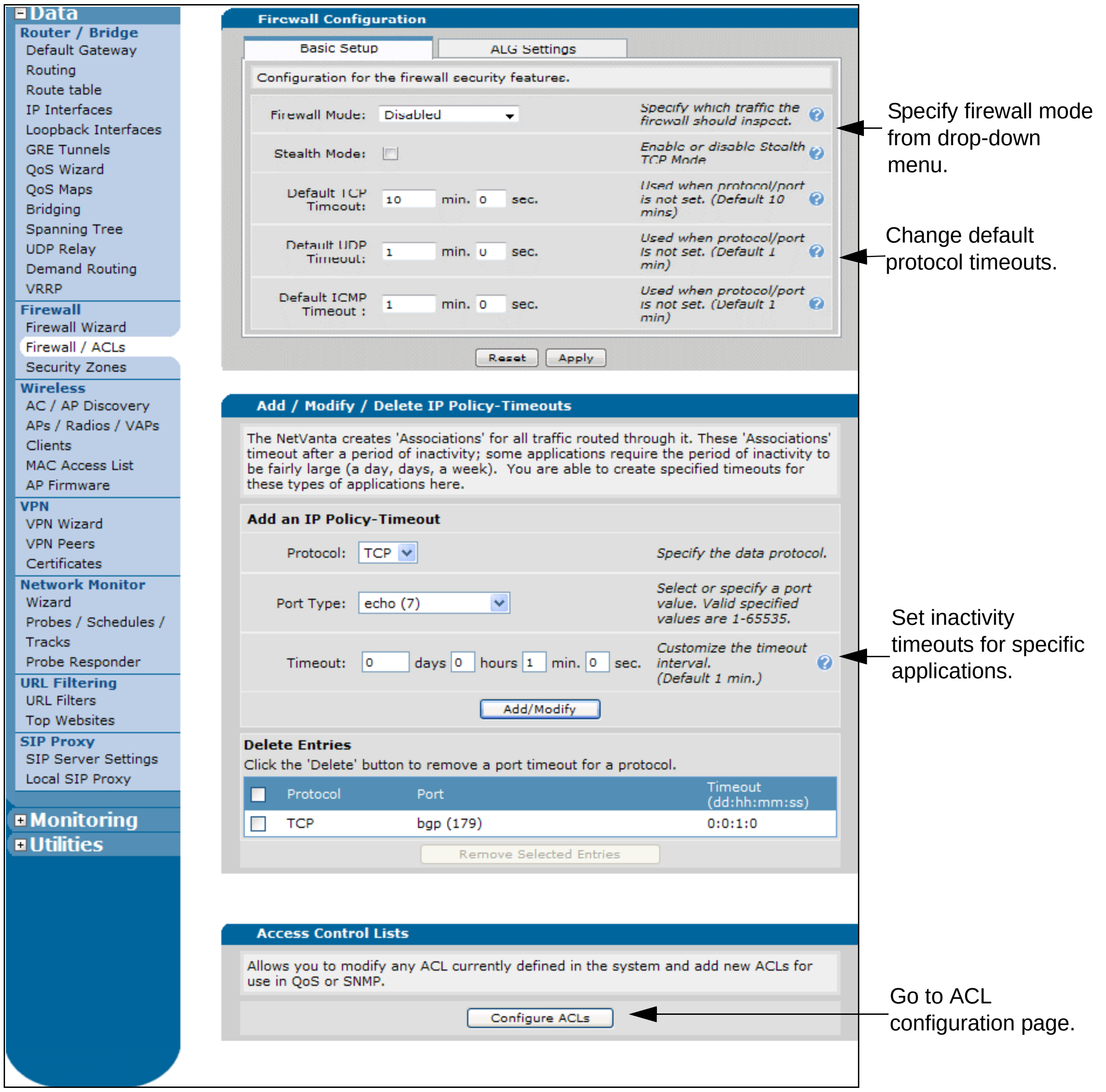
Firewall and ACLs

To configure IPv4 Firewall settings and ACLs, navigate to **Data** > **Firewall/ACLs**.

The **Firewall/ACL** menu is divided into three different sections. The **Firewall Configuration** section contains two tabs: the **Basic Setup** tab for basic IPv4 firewall setup options and the **ALG Settings** tab, which allows ALGs to be enabled or disabled easily. You can specify the firewall mode on the **Basic Setup** tab using the **Firewall Mode** drop-down menu, selecting either **Disabled**, **All traffic**, or **Local Traffic Only**.

The **Add/Modify/Delete IP Policy-Timeouts** section is where timeout intervals can be customized for TCP, UDP, and ICMP protocols. In addition, the port type can be used to customize the timeout intervals for specific applications. The drop-down menu for port type makes it easy to identify a service by either name or port number.

At the bottom, the **Access Control Lists** section accesses a menu that allows modification of any ACL currently defined in the system and the addition of new ACLs.



Firewall Configuration

Basic Setup | ALG Settings

Configuration for the firewall security features.

Firewall Mode:	Disabled	Specify which traffic the firewall should inspect.
Stealth Mode:	☐	Enable or disable Stealth TCP Mode
Default TCP Timeout:	10 min. 0 sec.	Used when protocol/port is not set. (Default 10 mins)
Default UDP Timeout:	1 min. 0 sec.	Used when protocol/port is not set. (Default 1 min)
Default ICMP Timeout :	1 min. 0 sec.	Used when protocol/port is not set. (Default 1 min)

Reset Apply

Add / Modify / Delete IP Policy-Timeouts

The NetVanta creates 'Associations' for all traffic routed through it. These 'Associations' timeout after a period of inactivity; some applications require the period of inactivity to be fairly large (a day, days, a week). You are able to create specified timeouts for these types of applications here.

Add an IP Policy-Timeout

Protocol:	TCP	Specify the data protocol.
Port Type:	echo (7)	Select or specify a port value. Valid specified values are 1-65535.
Timeout:	0 days 0 hours 1 min. 0 sec.	Customize the timeout interval. (Default 1 min.)

Add/Modify

Delete Entries

Click the 'Delete' button to remove a port timeout for a protocol.

☐	Protocol	Port	Timeout (dd:hh:mm:ss)
☐	TCP	bgp (179)	0:0:1:0

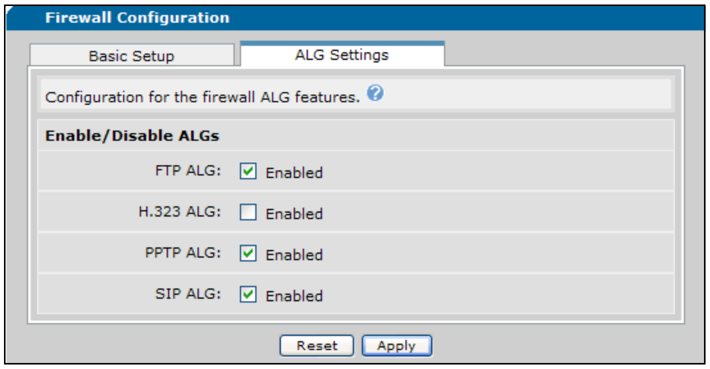
Remove Selected Entries

Access Control Lists

Allows you to modify any ACL currently defined in the system and add new ACLs for use in QoS or SNMP.

Configure ACLs

The **ALG Settings** tab lists four ALGs that can be easily enabled or disabled: FTP, H.323, PPTP, and SIP.


Firewall Configuration

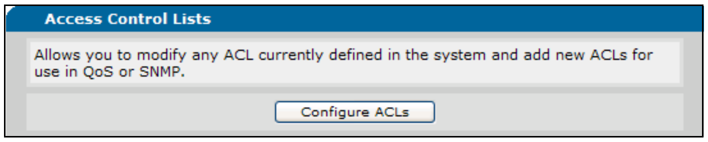
Configuration for the firewall ALG features.

Enable/Disable ALGs

FTP ALG:	✓ Enabled
H.323 ALG:	☐ Enabled
PPTP ALG:	✓ Enabled
SIP ALG:	✓ Enabled

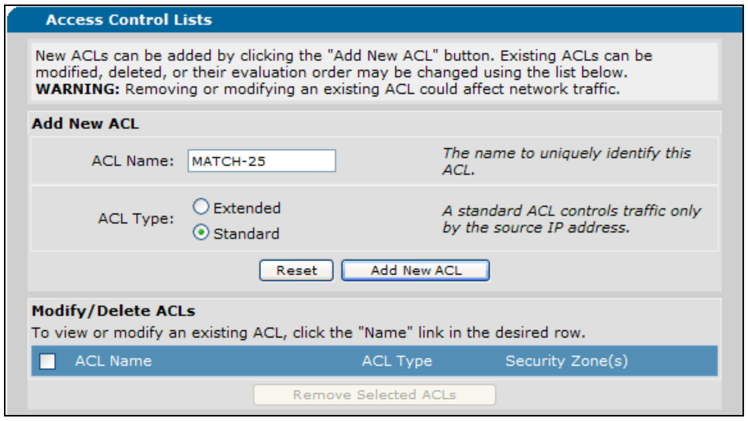
Configure ACLs

Select **Configure ACLs** at the bottom of the Firewall/ACL menu to access the ACL configuration menu.


Access Control Lists

Allows you to modify any ACL currently defined in the system and add new ACLs for use in QoS or SNMP.

Create a new ACL by entering a name and selecting either the standard or extended ACL Type. will be standard or extended **ACL Type**. A **Standard ACL** only filters based on packet source IPv4 address or hostname. An **Extended ACL** filters based on protocol, source and destination information, and TCP or UDP port numbers. Select **Add New ACL**.


Access Control Lists

New ACLs can be added by clicking the "Add New ACL" button. Existing ACLs can be modified, deleted, or their evaluation order may be changed using the list below.
WARNING: Removing or modifying an existing ACL could affect network traffic.

Add New ACL

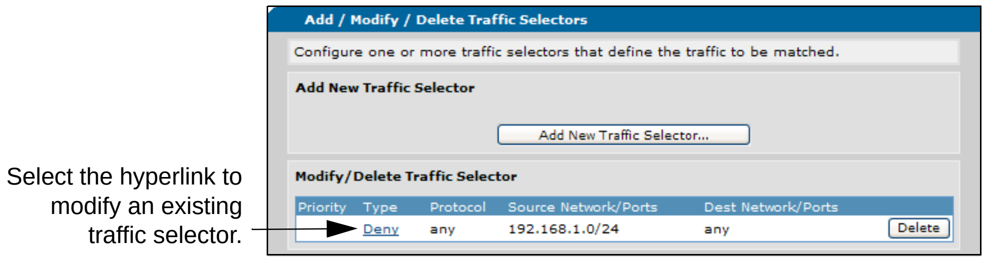
ACL Name:	MATCH-25	*The name to uniquely identify this ACL.*
ACL Type:	Extended [✓ Standard]	*A standard ACL controls traffic only by the source IP address.*

Modify/Delete ACLs

To view or modify an existing ACL, click the "Name" link in the desired row.

☐	ACL Name	ACL Type	Security Zone(s)

If there are no existing traffic selector entries within the ACL or if a new or additional traffic selector needs to be generated, select **Add New Traffic Selector**. To modify an existing traffic selector select a hyperlink in the **Type** column.


Select the hyperlink to modify an existing traffic selector.
Add / Modify / Delete Traffic Selectors
Add New Traffic Selector...
Delete

Configure one or more traffic selectors that define the traffic to be matched.

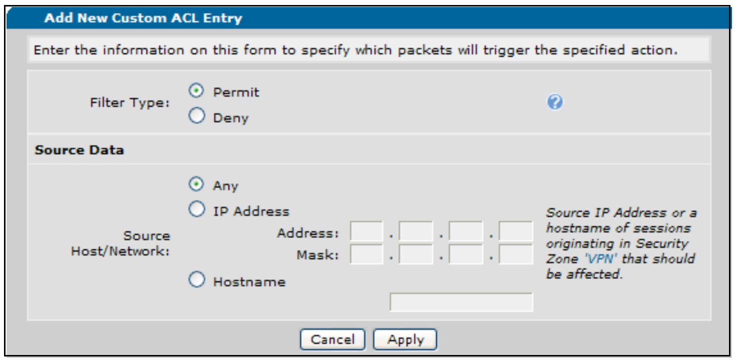
Add New Traffic Selector

Modify/Delete Traffic Selector

Priority	Type	Protocol	Source Network/Ports	Dest Network/Ports
	Deny	any	192.168.1.0/24	any

Standard ACL Entry Page

The standard ACL configuration menu contains fields to specify the filter type and the source data. Select the radio button next to **Permit** if traffic matching the criteria is to be processed or select the radio button next to **Deny** if traffic matching the criteria is not to be processed any further. Next, specify whether the packet source will come from any IPv4 address, a specific IPv4 address, or the hostname can be specified. Select **Apply**. When all traffic selector entries have been completed, go to *Change Order of Entries on page 47*.



Enter the information on this form to specify which packets will trigger the specified action.

Filter Type: [✓ Permit] Deny

Source Data

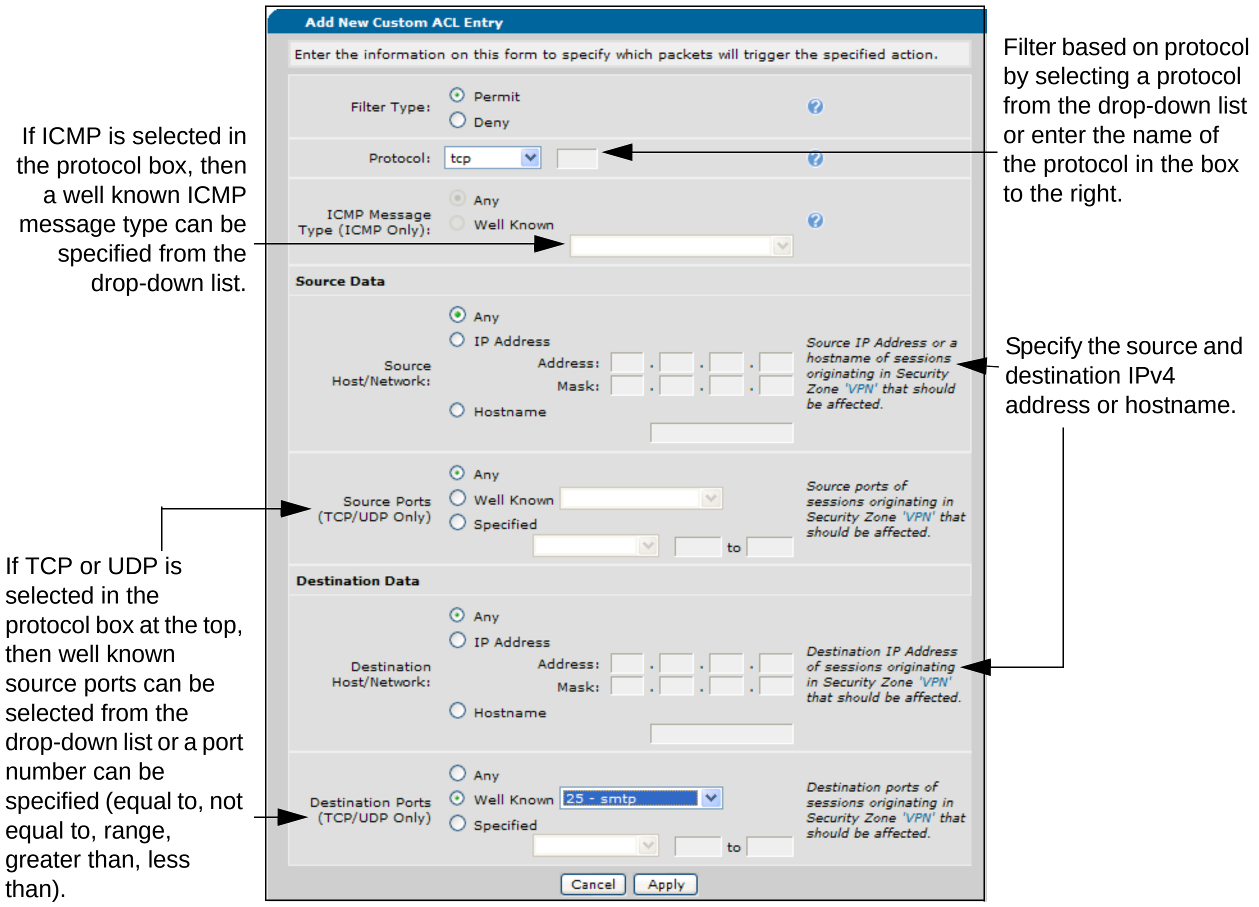
Source Host/Network: [✓ Any] IP Address Address: . . . Mask: . . . Hostname

Source IP Address or a hostname of sessions originating in Security Zone 'VPN' that should be affected.

NOTE *Multiple traffic selector entries can be added to a single ACL.*

Extended ACL Entry Page

The extended ACL menu configuration contains numerous fields with filtering options. Select the radio button next to **Permit** if traffic matching the criteria is to be processed or select the radio button next to **Deny** if traffic matching the criteria is not to be processed any further. Traffic can be filtered by protocol, various source and destination information, and TCP or UDP port numbers. Refer to *Create an IPv4 ACL on page 19* for detailed information on filtering options for extended ACLs. When all traffic selector entries have been completed, go to *Change Order of Entries on page 47*



If ICMP is selected in the protocol box, then a well known ICMP message type can be specified from the drop-down list.

If TCP or UDP is selected in the protocol box at the top, then well known source ports can be selected from the drop-down list or a port number can be specified (equal to, not equal to, range, greater than, less than).

Filter based on protocol by selecting a protocol from the drop-down list or enter the name of the protocol in the box to the right.

Specify the source and destination IPv4 address or hostname.

Add New Custom ACL Entry

Enter the information on this form to specify which packets will trigger the specified action.

Filter Type: [✓ Permit] Deny

Protocol: tcp

ICMP Message Type (ICMP Only): [✓ Any] Well Known

Source Data

Source Host/Network: [✓ Any] IP Address (Address: . . . ; Mask: . . .) Hostname

Source IP Address or a hostname of sessions originating in Security Zone 'VPN' that should be affected.

Source Ports (TCP/UDP Only): [✓ Any] Well Known Specified to

Source ports of sessions originating in Security Zone 'VPN' that should be affected.

Destination Data

Destination Host/Network: [✓ Any] IP Address (Address: . . . ; Mask: . . .) Hostname

Destination IP Address of sessions originating in Security Zone 'VPN' that should be affected.

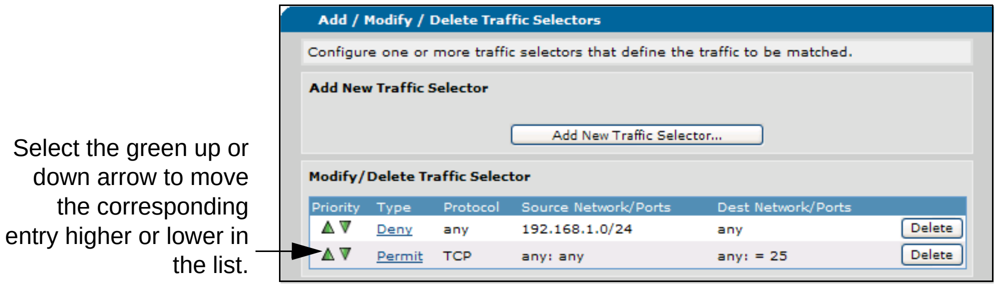
Destination Ports (TCP/UDP Only): Any [✓ Well Known] 25 - smtp Specified to

Destination ports of sessions originating in Security Zone 'VPN' that should be affected.

NOTE *Multiple traffic selector entries can be added to a single ACL.*

Change Order of Entries

The order of entries within an ACL to be easily changed without having to remove any of the entries from the ACL. The order of entries is crucial since the ACL will match traffic based on criteria from the top down. Move an entry up or down by selecting on the corresponding green arrow next to the statement you want to move.


Select the green up or down arrow to move the corresponding entry higher or lower in the list.
Add / Modify / Delete Traffic Selectors
Add New Traffic Selector
Add New Traffic Selector...
Modify/Delete Traffic Selector
Delete
Delete

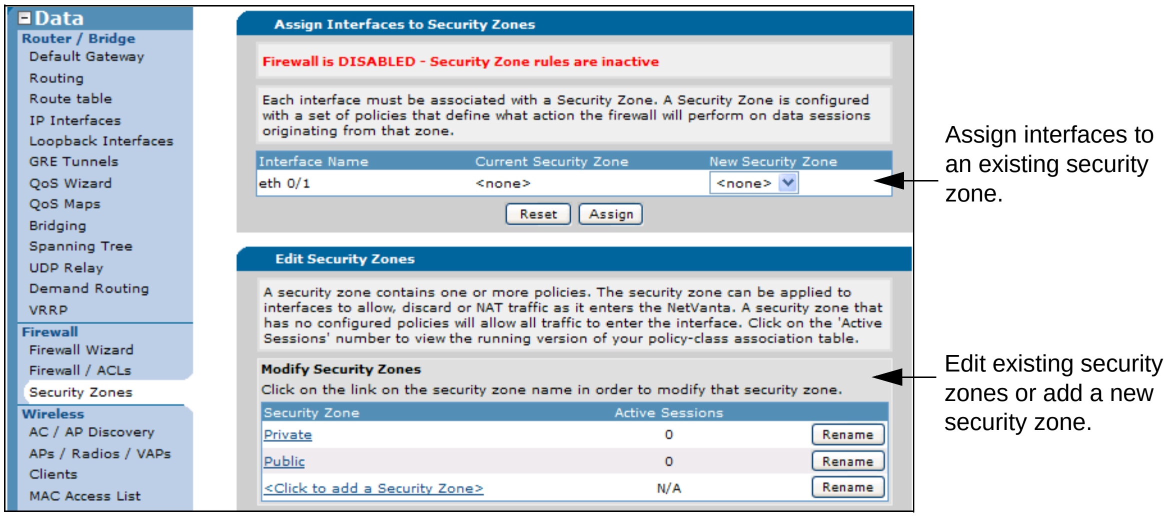
Configure one or more traffic selectors that define the traffic to be matched.

Priority	Type	Protocol	Source Network/Ports	Dest Network/Ports	
▲▼	Deny	any	192.168.1.0/24	any	Delete
▲▼	Permit	TCP	any: any	any: = 25	Delete

> NOTE *Remember to save the configuration to ensure the settings will not be lost after a restart. To save the configuration select **Save** in the upper right-hand corner of the GUI heading.*

Security Zones

To configure security zones, navigate to **Data** > **Firewall** > **Security Zones**. The Security Zones menu can be used to configure advanced firewall options that cannot be set up in the Firewall Wizard. A security zone contains one or more policies and can be applied to interfaces to allow, discard, or NAT traffic as it enters the AOS device.



Firewall is DISABLED - Security Zone rules are inactive

Each interface must be associated with a Security Zone. A Security Zone is configured with a set of policies that define what action the firewall will perform on data sessions originating from that zone.

Interface Name	Current Security Zone	New Security Zone
eth 0/1	<none>	<none>

A security zone contains one or more policies. The security zone can be applied to interfaces to allow, discard or NAT traffic as it enters the NetVanta. A security zone that has no configured policies will allow all traffic to enter the interface. Click on the 'Active Sessions' number to view the running version of your policy-class association table.

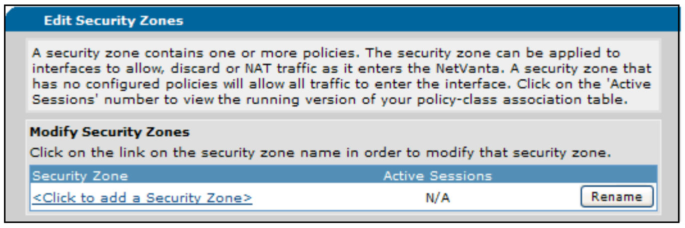
Click on the link on the security zone name in order to modify that security zone.

Security Zone	Active Sessions	
Private	0	Rename
Public	0	Rename
<Click to add a Security Zone>	N/A	Rename

> NOTE *A security zone that has no configured policies will allow all traffic to enter the interface(s) assigned to it. If the firewall is configured to process local traffic only, security zone rules will be inactive for forwarded traffic.*

Create a New Security Zone

Add a new security zone by selecting the **<Click to add a Security Zone>** hyperlink.



Edit Security Zones

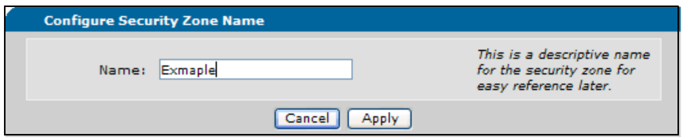
A security zone contains one or more policies. The security zone can be applied to interfaces to allow, discard or NAT traffic as it enters the NetVanta. A security zone that has no configured policies will allow all traffic to enter the interface. Click on the 'Active Sessions' number to view the running version of your policy-class association table.

Modify Security Zones

Click on the link on the security zone name in order to modify that security zone.

Security Zone	Active Sessions	
<Click to add a Security Zone>	N/A	Rename

Enter in a name for the security zone. The name *Example* is used here as an example. Select **Apply**.

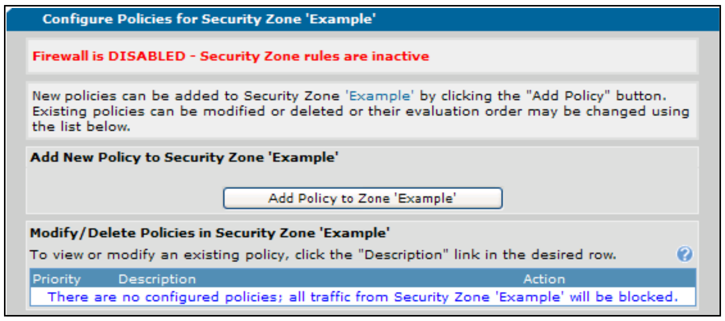


Configure Security Zone Name

Name: Exmaple

This is a descriptive name for the security zone for easy reference later.

At least one policy must be added to the security zone. Otherwise, if no policies are configured for the security zone, then the zone not will allow all traffic to enter the interface(s) assigned to it. Select **Add Policy to Zone**.



Configure Policies for Security Zone 'Example'

Firewall is DISABLED - Security Zone rules are inactive

New policies can be added to Security Zone 'Example' by clicking the "Add Policy" button. Existing policies can be modified or deleted or their evaluation order may be changed using the list below.

Add New Policy to Security Zone 'Example'

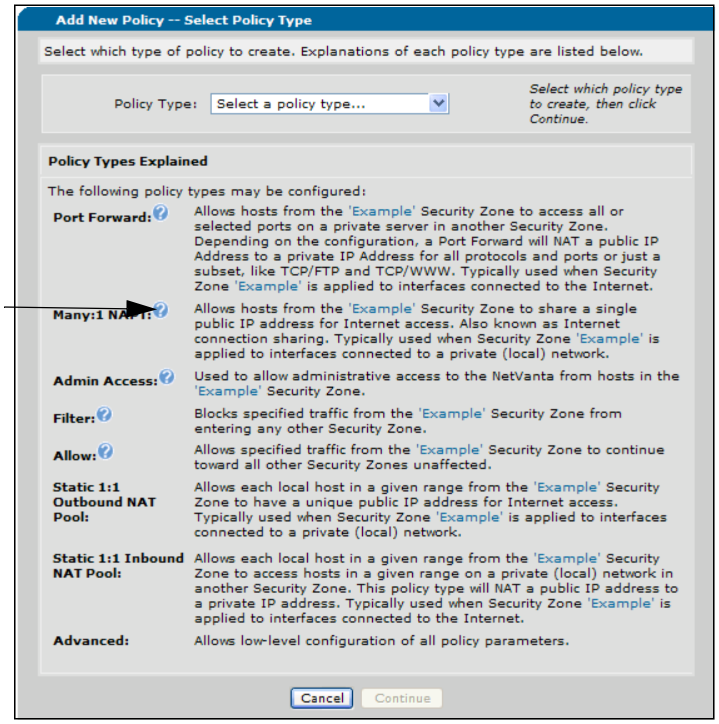
Modify/Delete Policies in Security Zone 'Example'

To view or modify an existing policy, click the "Description" link in the desired row.

Priority	Description	Action
	There are no configured policies; all traffic from Security Zone 'Example' will be blocked.	

Select the desired **Policy Type** from the drop-down list. A brief description of each policy type is displayed. Hover over the question mark beside a policy to display a diagram that aids in the understanding of the function of each individual policy type.

Hover over the question marks to see a diagram that aids in understanding the function of each individual policy type.



Add New Policy -- Select Policy Type

Select which type of policy to create. Explanations of each policy type are listed below.

Policy Type: Select a policy type...

Select which policy type to create, then click Continue.

Policy Types Explained

The following policy types may be configured:

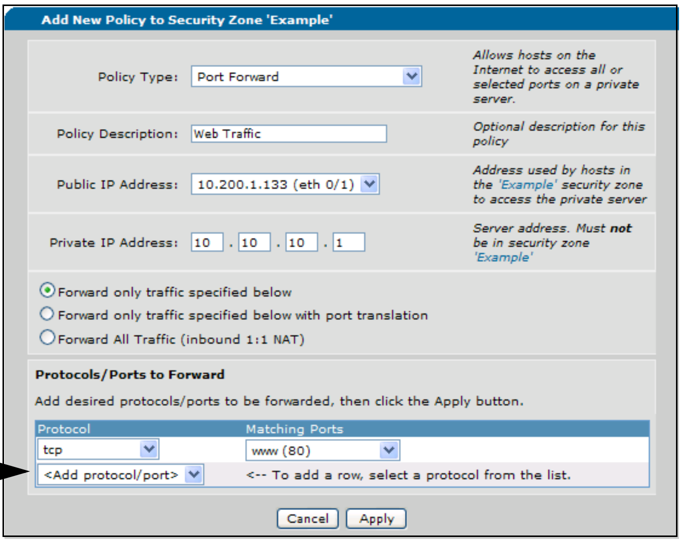
Port Forward:	Allows hosts from the 'Example' Security Zone to access all or selected ports on a private server in another Security Zone. Depending on the configuration, a Port Forward will NAT a public IP Address to a private IP Address for all protocols and ports or just a subset, like TCP/FTP and TCP/WWW. Typically used when Security Zone 'Example' is applied to interfaces connected to the Internet.
Many:1 NAPT:	Allows hosts from the 'Example' Security Zone to share a single public IP address for Internet access. Also known as Internet connection sharing. Typically used when Security Zone 'Example' is applied to interfaces connected to a private (local) network.
Admin Access:	Used to allow administrative access to the NetVanta from hosts in the 'Example' Security Zone.
Filter:	Blocks specified traffic from the 'Example' Security Zone from entering any other Security Zone.
Allow:	Allows specified traffic from the 'Example' Security Zone to continue toward all other Security Zones unaffected.
Static 1:1 Outbound NAT Pool:	Allows each local host in a given range from the 'Example' Security Zone to have a unique public IP address for Internet access. Typically used when Security Zone 'Example' is applied to interfaces connected to a private (local) network.
Static 1:1 Inbound NAT Pool:	Allows each local host in a given range from the 'Example' Security Zone to access hosts in a given range on a private (local) network in another Security Zone. This policy type will NAT a public IP address to a private IP address. Typically used when Security Zone 'Example' is applied to interfaces connected to the Internet.
Advanced:	Allows low-level configuration of all policy parameters.

Cancel Continue

Port Forwarding

The **Port Forward** policy type allows hosts from the selected security zone to access all or selected ports on a private server in another security zone. The example shown configures the IPv4 firewall to perform NAT only on HTTP traffic (port 80) destined for the public IP address 10.200.1.133 to the private IP address 10.10.10.1. Depending on the configuration, a port forward will NAT a public IPv4 address to a private IPv4 address for all protocols and ports or only a subset, like TCP/FTP and TCP/WWW. This is used when the selected security zone is applied to interfaces connected to the Internet.

Choose from the drop-down list to add another protocol/port to be forwarded. More than one protocol/port to forward may be specified.



Add New Policy to Security Zone 'Example'

Policy Type:	Port Forward	*Allows hosts on the Internet to access all or selected ports on a private server.*
Policy Description:	Web Traffic	*Optional description for this policy*
Public IP Address:	10.200.1.133 (eth 0/1)	*Address used by hosts in the 'Example' security zone to access the private server*
Private IP Address:	10 . 10 . 10 . 1	*Server address. Must **not** be in security zone 'Example'*

[✓ Forward only traffic specified below]
Forward only traffic specified below with port translation
Forward All Traffic (inbound 1:1 NAT)

Protocols/Ports to Forward

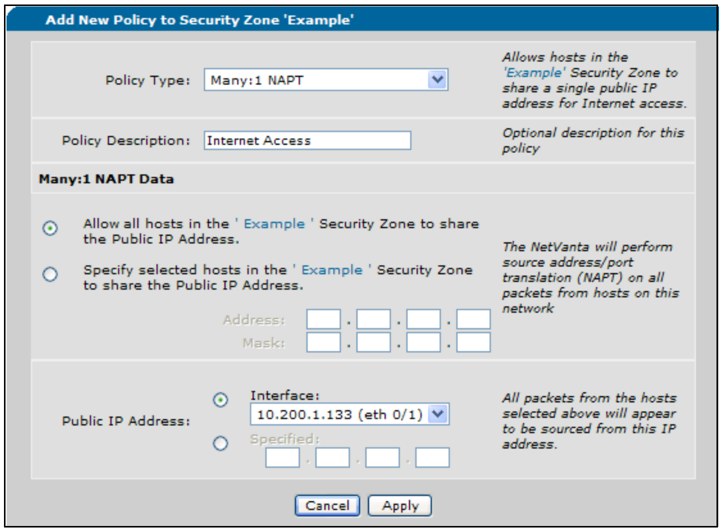
Add desired protocols/ports to be forwarded, then click the Apply button.

Protocol	Matching Ports
tcp	www (80)
<Add protocol/port>	<-- To add a row, select a protocol from the list.

Cancel Apply

Many:1 NAPT

The **Many:1 NAPT** policy type allows hosts from the selected security zone to share a single public IPv4 address for Internet access. All hosts or a subnet of hosts in the security zone can be selected to share the public IP address. Choose the public IPv4 address from configured public interfaces in the drop down menu or specify a specific IP address to be used. Many:1 NAPT is typically used when the selected security zone is applied to interfaces connected to an internal network. For additional information see *Many:1 NAT on page 14*.


Add New Policy to Security Zone 'Example'

Policy Type: Many:1 NAPT — *Allows hosts in the 'Example' Security Zone to share a single public IP address for Internet access.*

Policy Description: Internet Access — *Optional description for this policy*

Many:1 NAPT Data

(●) Allow all hosts in the ' Example ' Security Zone to share the Public IP Address.

( ) Specify selected hosts in the ' Example ' Security Zone to share the Public IP Address.

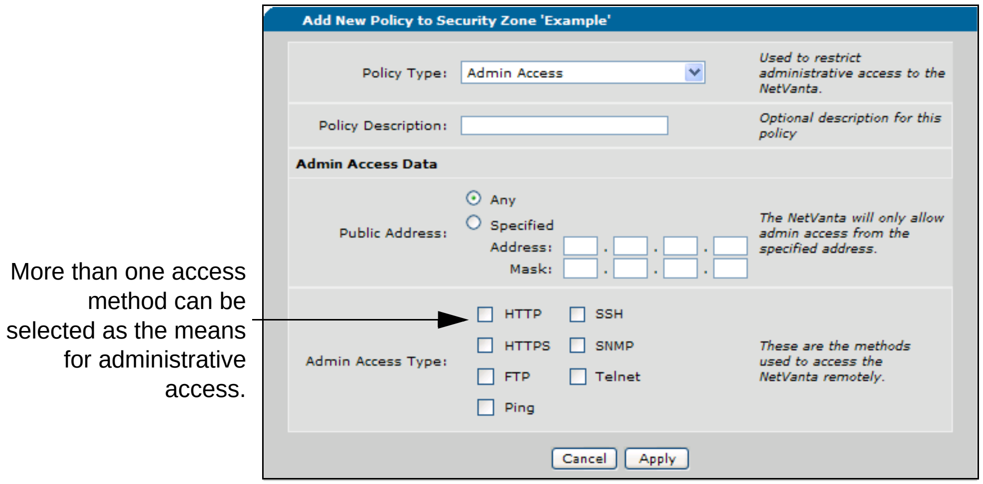
Address: . . .
Mask: . . .

The NetVanta will perform source address/port translation (NAPT) on all packets from hosts on this network

Public IP Address: (●) Interface: 10.200.1.133 (eth 0/1) ( ) Specified: . . . — *All packets from the hosts selected above will appear to be sourced from this IP address.*

Admin Access

The **Admin Access** policy type allows administrative access to the AOS device from hosts in the selected security zone. Access can be allowed from any host in the security zone or from a specific IPv4 address. Choose the method(s) allowed (for example, HTTP, Telnet, etc.) when the AOS device is accessed remotely for administration.


Add New Policy to Security Zone 'Example'

Policy Type: Admin Access — *Used to restrict administrative access to the NetVanta.*

Policy Description: — *Optional description for this policy*

Admin Access Data

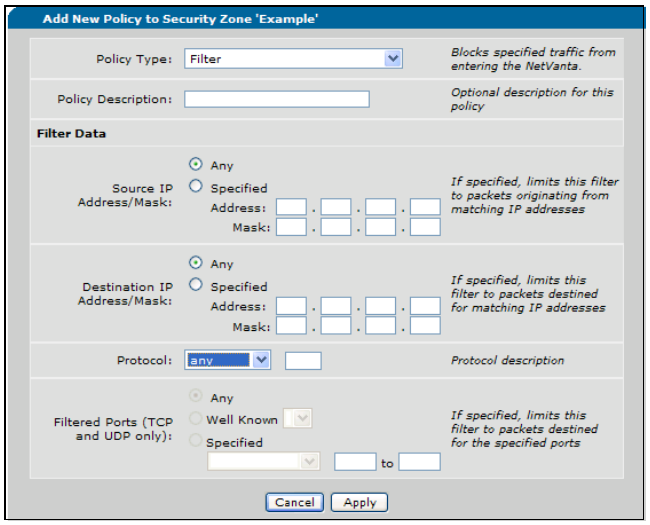
Public Address: (●) Any ( ) Specified Address: . . . Mask: . . . — *The NetVanta will only allow admin access from the specified address.*

Admin Access Type: ☐ HTTP ☐ SSH ☐ HTTPS ☐ SNMP ☐ FTP ☐ Telnet ☐ Ping — *These are the methods used to access the NetVanta remotely.*

Cancel
Apply
More than one access method can be selected as the means for administrative access.

Filter

The **Filter** policy type blocks specified traffic from this security zone from entering any other security zone. Data can be filtered by source IPv4 address, destination IPv4 address, and protocol. Additionally, the filter can be limited to packets destined for specific TCP and UDP ports.



Add New Policy to Security Zone 'Example'

Policy Type: Filter — *Blocks specified traffic from entering the NetVanta.*

Policy Description: — *Optional description for this policy*

Filter Data

Source IP Address/Mask: (●) Any ( ) Specified Address: . . . Mask: . . . — *If specified, limits this filter to packets originating from matching IP addresses*

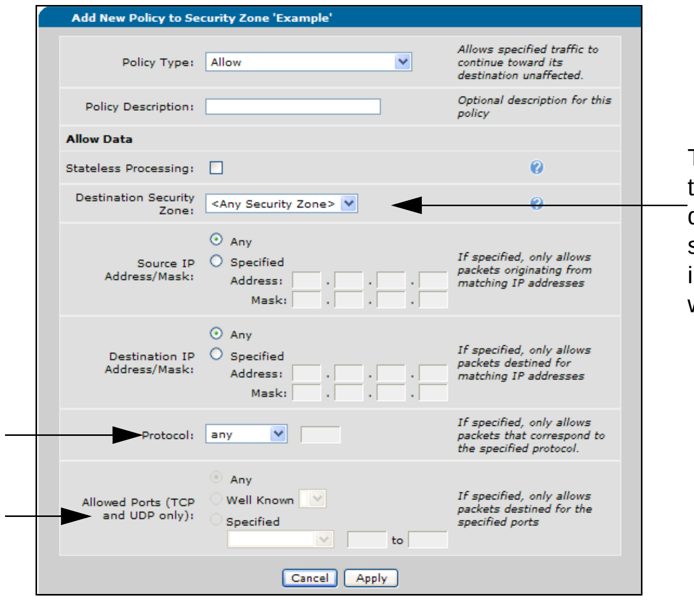
Destination IP Address/Mask: (●) Any ( ) Specified Address: . . . Mask: . . . — *If specified, limits this filter to packets destined for matching IP addresses*

Protocol: any — *Protocol description*

Filtered Ports (TCP and UDP only): Any, Well Known, Specified, to — *If specified, limits this filter to packets destined for the specified ports*

Allow

The **Allow** policy type permits specified traffic from this security zone to continue unaffected towards all other security zones or a specified destination security zone. By default, the IPv4 firewall performs stateful processing on packets. Select the box beside **Stateless Processing** to allow certain IP phones or point-of-sale (POS) stations to work in situations where stateful TCP processing prevents these devices from working. Data can be allowed by source IPv4 address, destination IPv4 address, and protocol. Additionally, the filter can be limited to packets destined for specific TCP and UDP ports.



Add New Policy to Security Zone 'Example'

Policy Type: Allow — *Allows specified traffic to continue toward its destination unaffected.*

Policy Description: — *Optional description for this policy*

Allow Data

Stateless Processing: ☐

Destination Security Zone: <Any Security Zone>

Source IP Address/Mask: (●) Any ( ) Specified Address: . . . Mask: . . . — *If specified, only allows packets originating from matching IP addresses*

Destination IP Address/Mask: (●) Any ( ) Specified Address: . . . Mask: . . . — *If specified, only allows packets destined for matching IP addresses*

Protocol: any — *If specified, only allows packets that correspond to the specified protocol.*

Allowed Ports (TCP and UDP only): Any, Well Known, Specified, to — *If specified, only allows packets destined for the specified ports*

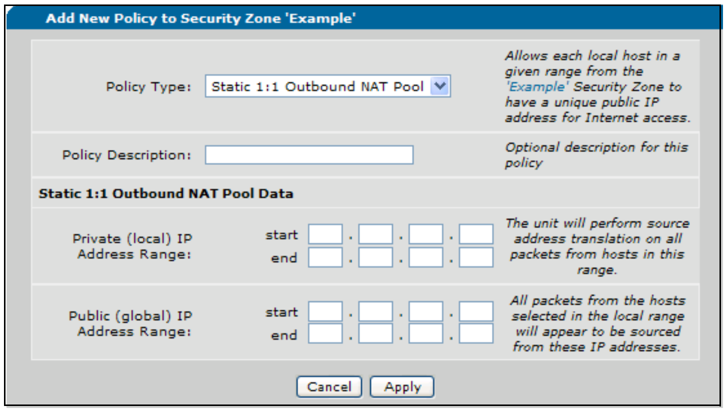
This option specifies that only traffic destined for the security zone selected in the drop-down menu will be allowed.

If the **TCP** or **UDP Protocol** is selected, then traffic can be further filtered by **Well Known** (chosen from the drop down menu) or specified ports.

Static 1:1 Outbound NAT Pool

The **Static 1:1 Outbound NAT Pool** policy type allows each local host in a given range from this security zone to have a unique public IPv4 address for Internet access. For every private host that requires a 1:1 NAT mapping, there must be a corresponding NAT address on the public side. This policy is typically used when the security zone is applied to interfaces connected to a private (local) network.

NOTE *Detailed discussion on NAT Pools is outside the scope of this document. Refer to the configuration guide NAT Pools in AOS (available online at https://supportforums.adtran.com) for more information on NAT pools.*



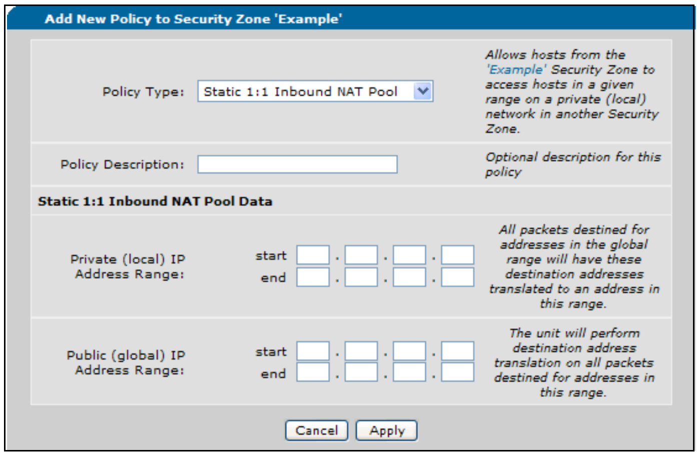
Add New Policy to Security Zone 'Example'

Policy Type:	Static 1:1 Outbound NAT Pool	*Allows each local host in a given range from the 'Example' Security Zone to have a unique public IP address for Internet access.*
Policy Description:		*Optional description for this policy*
Static 1:1 Outbound NAT Pool Data		
Private (local) IP Address Range:	start . . . end . . .	*The unit will perform source address translation on all packets from hosts in this range.*
Public (global) IP Address Range:	start . . . end . . .	*All packets from the hosts selected in the local range will appear to be sourced from these IP addresses.*

Static 1:1 Inbound NAT Pool

The **Static 1:1 Inbound NAT Pool** policy type allows each local host in a given range from this security zone to access hosts in a given range on a private (local) network in another security zone. This policy type will NAT a public IPv4 address to a private IPv4 address. Typically used when the security zone is applied to interfaces connected to the Internet.

NOTE *Detailed discussion on NAT pools is outside the scope of this document. Refer to the configuration guide NAT Pools in AOS (available online at https://supportforums.adtran.com) for more information on NAT pools.*

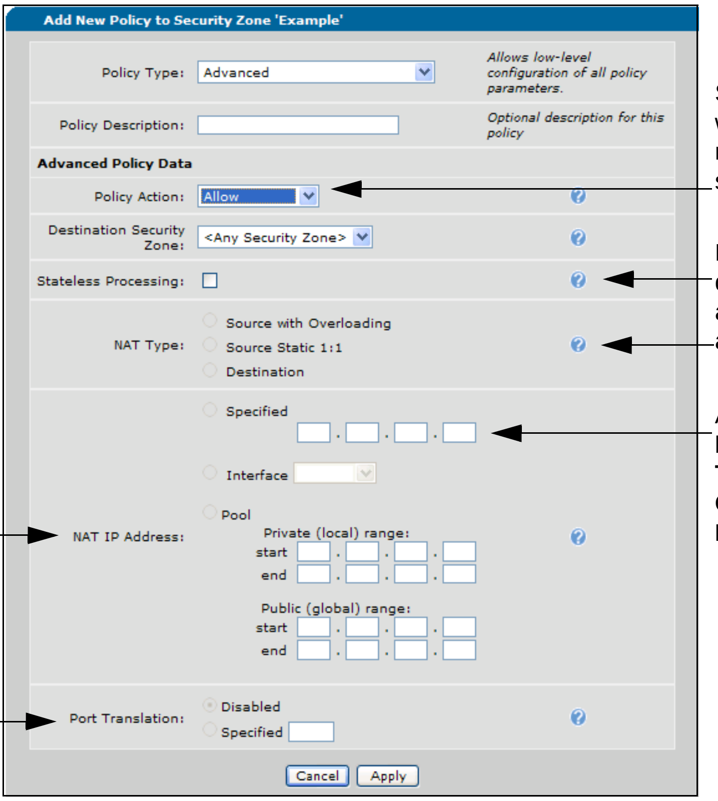


Add New Policy to Security Zone 'Example'

Policy Type:	Static 1:1 Inbound NAT Pool	*Allows hosts from the 'Example' Security Zone to access hosts in a given range on a private (local) network in another Security Zone.*
Policy Description:		*Optional description for this policy*
Static 1:1 Inbound NAT Pool Data		
Private (local) IP Address Range:	start . . . end . . .	*All packets destined for addresses in the global range will have these destination addresses translated to an address in this range.*
Public (global) IP Address Range:	start . . . end . . .	*The unit will perform destination address translation on all packets destined for addresses in this range.*

Advanced

The **Advanced** policy type allows low-level configuration of all policy parameters. Experienced administrators can use this type to configure settings for multiple facets of a policy all on one menu. Hover over the question marks to the right of the configurable parameters to view more information on how the setting will affect the policy.



Add New Policy to Security Zone 'Example'

Policy Type:	Advanced	*Allows low-level configuration of all policy parameters.*
Policy Description:		*Optional description for this policy*
Advanced Policy Data		
Policy Action:	Allow	
Destination Security Zone:	<Any Security Zone>	
Stateless Processing:	☐	
NAT Type:	○ Source with Overloading ○ Source Static 1:1 ○ Destination	
NAT IP Address:	○ Specified ○ Interface ○ Pool Private (local) range: start, end Public (global) range: start, end	
Port Translation:	● Disabled ○ Specified	

Select the action that will be taken if a packet matches the entries specified below.

Hover over the question marks to view additional information about the settings.

An IPv4 address can be specified with **NAT Type Source with Overloading** or **Destination**.

A **NAT IP Address Pool** can be specified with the **Source Static 1:1** and the **Destination NAT Type** options.

Port Translation can be enabled with the **Destination NAT Type** option.

NOTE *Depending on the **Policy Action** and the **NAT Type** selected, the radio button next to certain options are dimmed. A dimmed option means that it is not available for the current combination of selections.*

NOTE *When configuring security zone policies using the **Advanced** menu, it is important to remember to add the appropriate traffic selectors to the security zone policy. AOS automatically adds traffic selectors when a specific security zone type, such as **Many:1 NAPT**, is selected. However, AOS will NOT automatically add traffic selectors to policies created using the **Advanced** menu.*

The **Policy Action** is the action that will be taken if a packet matches the entries specified in the other fields on this form. Using the drop-down list choose **Allow**, **AllowReverse**, **Discard**, and **NAT**. The **Allow** option allows traffic that matches the traffic selector to be forwarded without modification. **AllowReverse** allows traffic that reverse-matches the traffic selector to be forwarded without modification. To perform the reverse-match, the source and destination information in the packet are swapped and compared with the traffic selector. The **AllowReverse** action is useful for a public-facing security zone, where the traffic selector is the same as a VPN selector. A straightforward **Allow** entry would be used on the private-facing interface, whereas the **AllowReverse** entry allows traffic initiated from the public-facing side to be forwarded without changing the traffic selector.

The **Destination Security Zone** specifies that the policy action will be taken only if the traffic is destined for the selected security zone. Using the drop-down list, choose **<Any Security Zone>**, **<Self Bound>** or an existing named security zone that appears in the list. The **Self Bound** selection allows packets if they are directed to the AOS device.

Select the box beside **Stateless Processing** to allow certain IP phones or POS stations to work in situations where stateful TCP processing prevents these devices from working. Data can be allowed by source IPv4 address, destination IPv4 address, and protocol. This option can only be enabled when the Policy Action is set to **Allow** or **AllowReverse** (NAT processing is always stateful).

If the **NAT** action is chosen, then the NAT Type and NAT IP Address fields must be specified. Choose the NAT type, **Source with Overloading**, **Source Static 1:1**, and **Destination** using the radio buttons.

If **Source with Overloading** is selected, the source address is replaced with the NAT IP Address. The source port will be translated if the port is in use by another session. A NAT IP address to replace the source address are configured as a **Specified** IPv4 address or an interface selected from the **Interface** drop-down menu.

If **Source Static 1:1** is selected, a source address from the **Private (local) range** in the **Pool** is replaced with the corresponding IPv4 address in the **Public (global) range**. IPv4 addresses or ranges for the **Pool** are configured by the user. No ports are translated.

If **Destination** is selected, then either a NAT IP address can be specified or a NAT IP address pool can be entered. The **Specified** IPv4 address replaces the destination IPv4 address with the specified NAT IP address.

NOTE *If an IPv4 address is specified, **Port Translation** can be enabled. Packets arriving from this security zone will have their destination port replaced with the specified port number.*

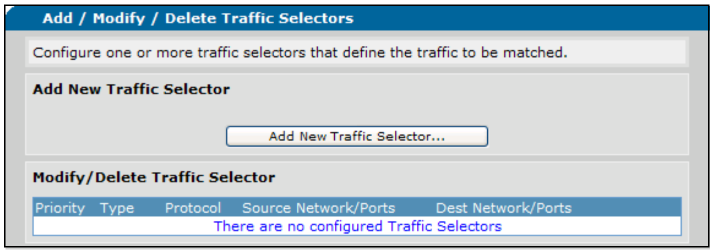
The **Pool** option replaces the destination address in the public (global) range with the corresponding address in the private (local) range. No ports are translated if **NAT IP address** pool is selected.

NOTE *Detailed discussion on NAT Pools is outside the scope of this document. Refer to the configuration guide NAT Pools in AOS (available online at https://supportforums.adtran.com) for more information on NAT pools.*

Select **Apply**.

Add, Modify, and Delete Traffic Selectors

When creating certain policies, a new traffic selector may need to be generated. If a new traffic selector must be added, select the **Add New Traffic Selector** at the bottom of the menu. The **Add new Custom ACL** entry menu will appear. Refer to *Extended ACL Entry Page on page 46* for detailed information.


Add / Modify / Delete Traffic Selectors

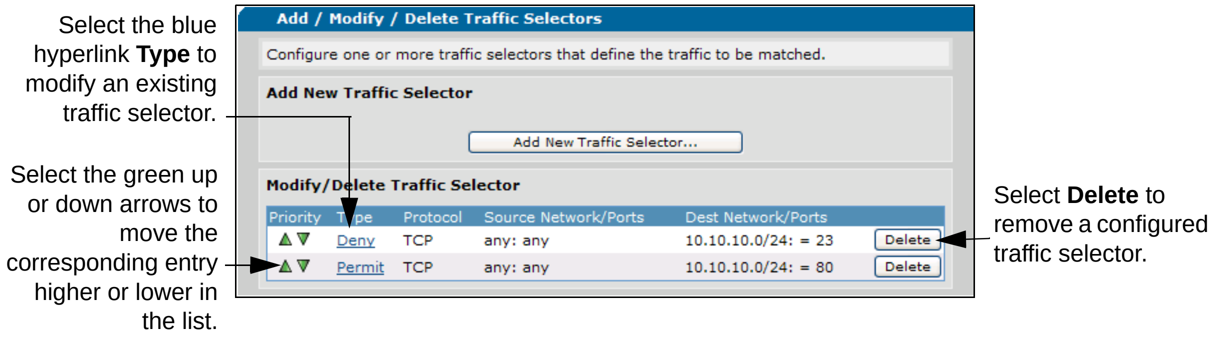
Configure one or more traffic selectors that define the traffic to be matched.

Add New Traffic Selector

Modify/Delete Traffic Selector

Priority	Type	Protocol	Source Network/Ports	Dest Network/Ports
There are no configured Traffic Selectors				

The **Modify/Delete Traffic Selector** section will also appear at the bottom of the menu for a policy with existing traffic selector entries. Select the blue hyperlink under the **Type** column to enter the configuration menu to modify an existing traffic selector. Select **Delete** to remove an existing traffic selector. Change the priority of an entry in the list by selecting the green up or down arrows next to the statement you would like to move.


Add / Modify / Delete Traffic Selectors

Configure one or more traffic selectors that define the traffic to be matched.

Add New Traffic Selector

Add New Traffic Selector...

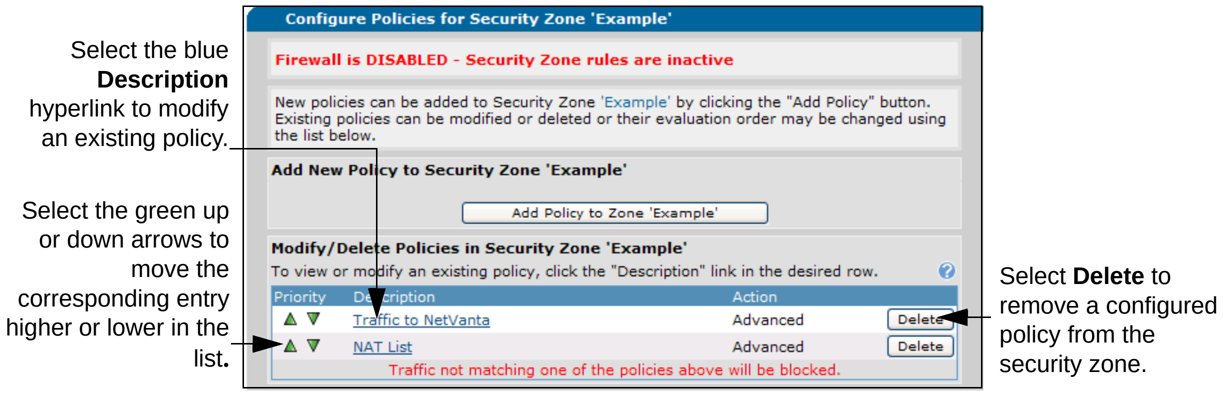
Modify/Delete Traffic Selector

Priority	Type	Protocol	Source Network/Ports	Dest Network/Ports	
▲▼	Deny	TCP	any: any	10.10.10.0/24: = 23	Delete
▲▼	Permit	TCP	any: any	10.10.10.0/24: = 80	Delete

Select the blue hyperlink **Type** to modify an existing traffic selector.
Select the green up or down arrows to move the corresponding entry higher or lower in the list.
Select **Delete** to remove a configured traffic selector.

Modify or Delete a Policy within a Security Zone

The modify/delete policy options for any existing security zone are located at the bottom of the menu. Assuming that at least one security policy has been configured for the security zone, select the blue policy name hyperlink under the **Type** column to enter the configuration menu to modify the existing policy. Remove a policy by selecting **Delete** to the right of the existing policy to be removed. Change the priority of an entry higher or lower in the list by selecting the corresponding green up or down arrows next to the statement.


Select the blue **Description** hyperlink to modify an existing policy.
Select the green up or down arrows to move the corresponding entry higher or lower in the list.
Select **Delete** to remove a configured policy from the security zone.
Configure Policies for Security Zone 'Example'
Add New Policy to Security Zone 'Example'
Add Policy to Zone 'Example'
Modify/Delete Policies in Security Zone 'Example'

Firewall is DISABLED - Security Zone rules are inactive

New policies can be added to Security Zone 'Example' by clicking the "Add Policy" button. Existing policies can be modified or deleted or their evaluation order may be changed using the list below.

To view or modify an existing policy, click the "Description" link in the desired row.

Priority	Description	Action	
▲ ▼	Traffic to NetVanta	Advanced	Delete
▲ ▼	NAT List	Advanced	Delete

Traffic not matching one of the policies above will be blocked.

*Remember to save the configuration to ensure the settings will not be lost after a restart. To save the configuration select **Save** in the upper right-hand corner of the GUI heading.*

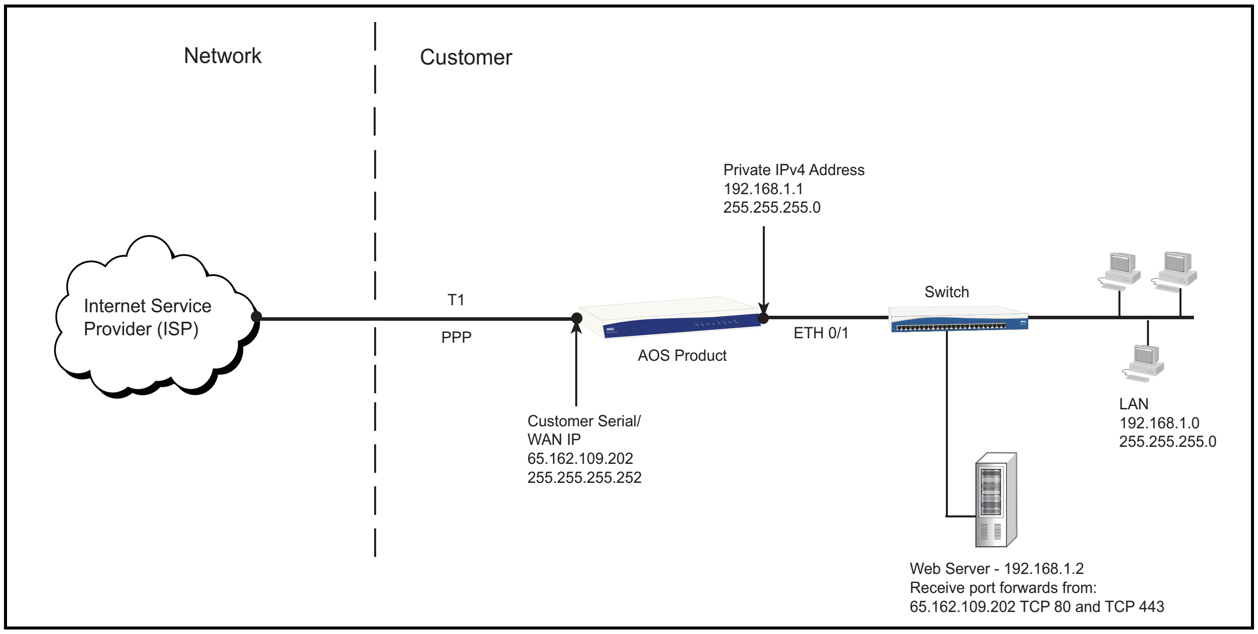
Example Configurations

Example 1 - Internet Access, Port Forward, Admin Access

In this example, an AOS router is providing Internet access for the 192.168.1.0 /24 subnet, as well as port forwarding HTTP and HTTPS to an internal Web server (192.168.1.2). Internet access is facilitated from the **nat source list** command in the **PRIVATE** ACP, which is applied to the local area network (LAN) interface of the unit (ETH 0/1). This statement replaces the source IPv4 address of hosts on the 192.168.1.0 /24 network with the wide area network (WAN) address assigned to the point-to-point protocol (PPP 1) interface (65.162.109.202) so their traffic can be routed on the Internet. The port forward is accomplished with the **nat destination list** command in the **PUBLIC** ACP, which is applied to the WAN interface (PPP 1). This NAT policy is executed for traffic that matches the referenced ACL (**WEB-ACL-4**), which only allows HTTP and HTTPS traffic destined to 65.162.109.202. Traffic matching this ACP will be directed to the internal Web server at 192.168.1.2. This causes a conflict with the built-in Web server on the AOS unit. To overcome this, the HTTP server is configured to run on TCP port 8080 and the HTTPS server is configured to run on TCP port 8081.

An administrative ACP has also been implemented that will only allow certain types of management traffic destined for the WAN interface on the unit. All traffic not matching this ACP, the port forward previously discussed, or return traffic from an open policy session on the **PRIVATE** ACP will be dropped. Allowing administrator access is accomplished with the **allow list WEB-ACL-3 self** command in the **PUBLIC** ACP. The referenced ACL (**WEB-ACL-3**) only accepts ping, SSH, TCP port 8080, and TCP port 8081 destined to the public IP of the unit (65.162.109.202). The SSH lines are enabled with the **no shutdown** command. Both the HTTPS server and the SSH lines use the local user list for authentication credentials. Users will need to use the user name **admin** and the password **password** as shown in the configuration.

Users on the private side of the unit will be able to access any management services that are configured and running, as they are allowed in from the **allow list self SELF** command in the **PRIVATE** ACP. This references an ACL that allows any IP traffic.



```
username "admin" password "password"
!
ip firewall
!
interface eth 0/1
  ip address  192.168.1.1  255.255.255.0
  ip access-policy PRIVATE
  no shutdown
!
interface t1 1/1
  clock source line
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface ppp 1
  ip address  65.162.109.202  255.255.255.252
  ip access-policy PUBLIC
  no shutdown
  cross-connect 1 t1 1/1 1 ppp 1
!
ip access-list standard WIZARD-ICS
  remark Internet Connection Sharing
  permit any
!
ip access-list extended SELF
  remark Traffic to AOS product
  permit ip any  any     log
!
ip access-list extended WEB-ACL-3
  remark Admin Access
  permit tcp any  host 65.162.109.202 eq 8080   log
  permit tcp any  host 65.162.109.202 eq 8081   log
  permit tcp any  host 65.162.109.202 eq ssh   log
  permit icmp any  host 65.162.109.202 echo   log
!
ip access-list extended WEB-ACL-4
  remark Web Server
  permit tcp any  host 65.162.109.202 eq www   log
  permit tcp any  host 65.162.109.202 eq https   log
!
ip policy-class PRIVATE
  allow list SELF self
  nat source list WIZARD-ICS interface ppp 1 overload
!
ip policy-class PUBLIC
  nat destination list WEB-ACL-4 address 192.168.1.2
  allow list WEB-ACL-3 self
!
ip route 0.0.0.0 0.0.0.0 ppp 1
!
ip http server 8080
ip http secure-server 8081
!
line ssh 0 4
  login local-userlist
  no shutdown
!
```

The command ***clock source line*** *is enabled by default. Therefore, this command will not appear in the output when the* ***show running-config*** *command is issued.*

NOTE

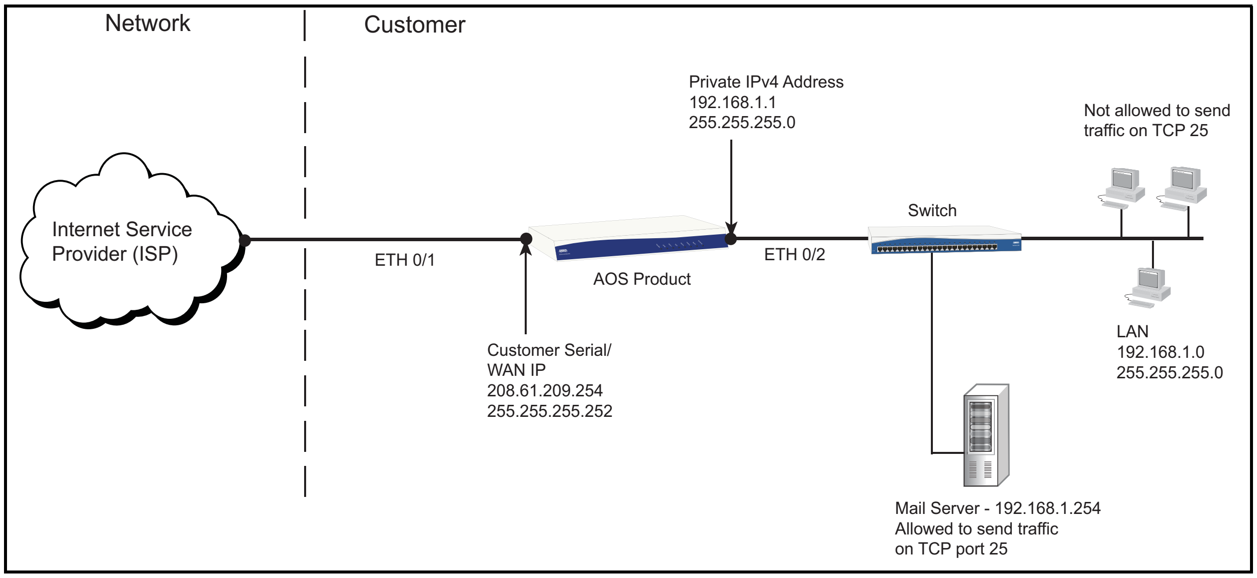
*Most of this configuration can be implemented easily in the AOS GUI by first running the Firewall Wizard and then navigating to **Data** > **Firewall** > **Security Zones**. Select **Public** in the **Edit Security Zones** menu. Select **Add Policy to Zone 'Public'**. Select **Port Forward** as the **Policy Type** from the drop-down menu. Select **Continue** at the bottom of the page and complete the configuration by populating the related fields. Once completed, move the newly created **Port Forward** to the top of the list with the arrows. More specific firewall rules should be placed above more general rules.*

Example 2 - Filtering Ports Outbound

In a network environment, most administrators have concerns regarding outbound user traffic. Sometimes it is necessary to prevent users inside the network from accessing a specific service on the Internet. At other times, to prevent spreading viruses, only specific servers should be allowed outbound on specific ports.

This example covers a common scenario, preventing any host, other than a mail server, from sending traffic outbound on port 25. To accomplish this, an additional firewall rule must be created. The first rule in the ACP named **PRIVATE** discards any traffic that matches the ACL named **MATCH-25**. Because **MATCH-25** was defined to first deny (not process) all traffic from the mail server and then permit (process) all simple mail transfer protocol (SMTP) traffic (port 25), all SMTP traffic not from the mail server is dropped by the **discard list** command in the ACP. The NAT for all other network traffic is specified to match everything else. Users on the private side of the unit will be able to access any management services that are configured and running, as they are allowed in from the **allow list SELF self** command in the **PRIVATE** ACP. This references an ACL that allows any IP traffic.

The **PUBLIC** ACP is empty so it will deny all incoming traffic from the Internet unless it is in response to traffic initiated from hosts on one of the LANs present in the firewall session table.



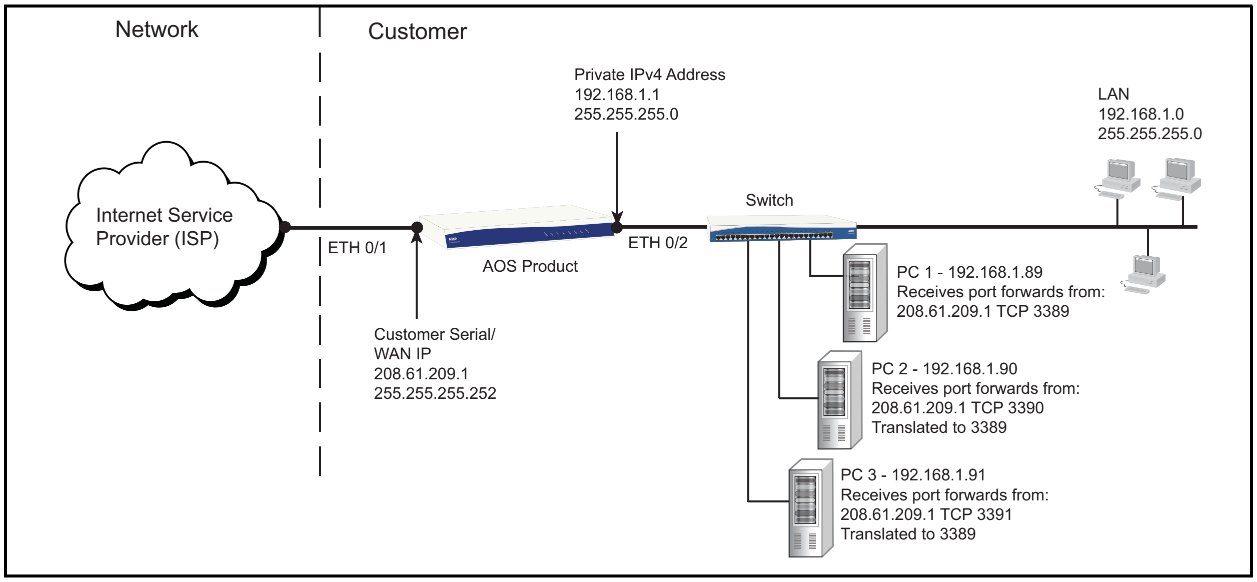
```
!
ip firewall
!
interface eth 0/1
```

```
  ip address  208.61.209.254  255.255.255.252
  ip access-policy PUBLIC
  no shutdown
!
interface eth 0/2
  ip address  192.168.1.1  255.255.255.0
  ip access-policy PRIVATE
  no shutdown
!
ip access-list extended MATCH-25
  deny ip host 192.168.1.254 any
  permit tcp any any eq smtp
!
ip access-list standard WIZARD-ICS
  remark Internet Connection Sharing
  permit any
!
ip access-list extended SELF
  remark Traffic to AOS product
  permit ip any any log
!
ip policy-class PRIVATE allow list SELF self
  discard list MATCH-25
  nat source list WIZARD-ICS interface eth 0/1 overload
!
ip policy-class PUBLIC
!Implicit discard
!
ip route 0.0.0.0 0.0.0.0 208.61.209.253
!
```

Example 3 - Port Translation for Multiple Hosts

In this example, port translation allows multiple hosts to support the same service through the firewall. Under normal circumstances, if a customer has only one public IP address, they can forward a single port (for example, TCP port 3389) only once. So in this instance, if there were multiple personal computers (PCs) attempting to use Microsoft Remote Desktop Protocol (RDP) on TCP port 3389, only one of them would function. One way around this is to configure port translation. Port translation allows a user to change a destination port as it passes through the firewall.

In this example, there are three PCs behind the firewall using Microsoft RDP. The PCs are addressed 192.168.1.89, 192.168.1.90, and 192.168.1.91. Since TCP port 3389 can only be used once, it is forwarded to the first PC. For the second and third PCs, the destination port in the ACLs defining the port forwarding is changed (ACL **RDP3390** and ACL **RDP3391**). This port represents the port to which an external user will actually attempt to connect. For example, port 3390 instead of 3389. Adding the extra parameter (**port 3389**) to the **nat destination list** commands in the **PUBLIC** ACP causes the router to change the destination port in transit. The end result is a user attempting to access 192.168.1.90 from the outside on TCP port 3390, appears to the PC to be using destination port 3389. In this manner, multiple PCs can support the same service. Users on the private side of the unit will be able to access any management services that are configured and running, as they are allowed in from the **allow list SELF self** command in the **PRIVATE** ACP. This references an ACL that allows any IP traffic.



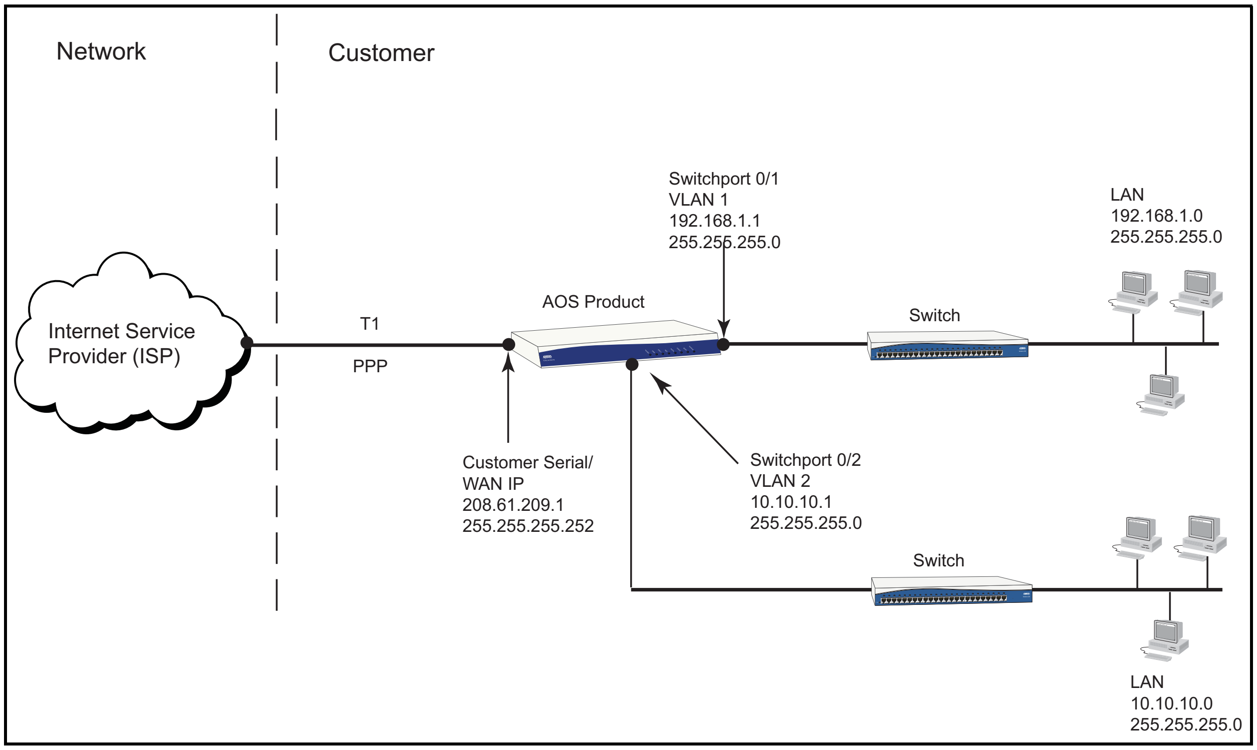
```
!
ip firewall
!
interface eth 0/1
  ip address  208.61.209.1  255.255.255.252
  ip access-policy PUBLIC
  no shutdown
!
interface eth 0/2
  ip address  192.168.1.1  255.255.255.0
  ip access-policy PRIVATE
  no shutdown
!
ip access-list standard WIZARD-ICS
  remark Internet Connection Sharing
  permit any
!
ip access-list extended RDP3389
  permit tcp any  any eq 3389
!
ip access-list extended RDP3390
  permit tcp any  any eq 3390
!
ip access-list extended RDP3391
  permit tcp any  any eq 3391
!
ip access-list extended SELF
  remark Traffic to AOS product
  permit ip any any log
!
ip policy-class Private
  allow list SELF self
  nat source list WIZARD-ICS interface eth 0/1 overload
!
ip policy-class PUBLIC
  nat destination list RDP3389 address 192.168.1.89
  nat destination list RDP3390 address 192.168.1.90 port 3389
  nat destination list RDP3391 address 192.168.1.91 port 3389
!
ip route 0.0.0.0 0.0.0.0 208.61.209.2
!
```

Example 4 - Stateless Allow Between LAN Interfaces

In an environment where a network is segmented into multiple private subnets, either by virtual local area networks (VLANs) or multiple physical interfaces, devices in separate subnets may need to communicate with one another. The default configuration installed through the AOS GUI Firewall Wizard will NAT all traffic, including traffic from LAN to LAN. While some applications might allow this communication, many will not. In this case, an **allow** ACP must be created to permit the private subnets to communicate without NAT.

In this example, an AOS unit is configured with two private VLANs, each of which is using the **PRIVATE** ACP. Since the two subnets are dissimilar and cannot be supernetted, an ACL must be created with two statements. Each ACL statement permits traffic in one direction. The ACL is then applied to the **PRIVATE** ACP as an **allow**. The **stateless** keyword is an optional parameter for the **allow list** command that prevents firewall timeouts, attack checks, and ALGs from tampering with traffic. Finally, since the ACPs are executed in sequential order, the **allow** statement must be placed above the NAT statement. This will catch all LAN-to-LAN traffic and prevent translating the network address. Users on the private side of the unit will be able to access any management services that are configured and running, as they are allowed in from the **allow list SELF self** command in the **PRIVATE** ACP. This references an ACL that allows any IP traffic to self.

Only HTTPS and SSH administrative access are allowed on the **PUBLIC** ACP because it references the **ADMIN-ACCESS** ACL.



```
!
ip firewall
!
!
interface switchport 0/1
  no shutdown
!
interface switchport 0/2
  no shutdown
  switchport access vlan 2
!
interface vlan 1
  ip address  192.168.1.1  255.255.255.0
  ip access-policy PRIVATE
  no shutdown
!
interface vlan 2
  ip address  10.10.10.1  255.255.255.0
  ip access-policy PRIVATE
  no shutdown
!
interface t1 1/1
  clock source line
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface ppp 1
  ip address  208.61.209.1  255.255.255.252
  ip access-policy PUBLIC
  no shutdown
  cross-connect 1 t1 1/1 1 ppp 1
!
ip access-list standard WIZARD-ICS
  remark Internet Connection Sharing
  permit any
!
ip access-list extended ADMIN-ACCESS
  permit tcp any any eq ssh
  permit tcp any any eq https
!
ip access-list extended INTER-VLAN
  permit ip 192.168.1.0 0.0.0.255 10.10.10.0 0.0.0.255
  permit ip 10.10.10.0 0.0.0.255 192.168.1.0 0.0.0.255
!
ip access-list extended SELF
  remark Traffic to AOS product
  permit ip any any log
!
ip policy-class PRIVATE
  allow list SELLF self
  allow list INTER-VLAN stateless
  nat source list WIZARD-ICS interface ppp 1 overload
!
ip policy-class PUBLIC
  allow ADMIN-ACCESS self
!
ip route 0.0.0.0 0.0.0.0 208.61.209.2
!
```

NOTE

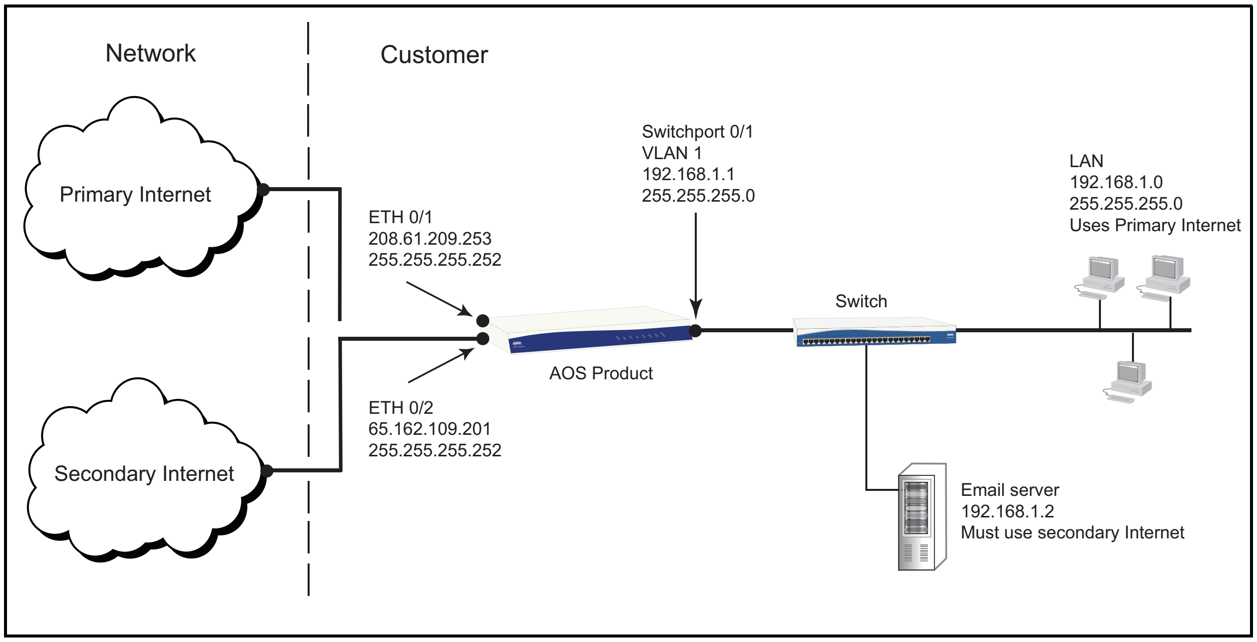
*The command **clock source line** is enabled by default. Therefore, this command will not appear in the output when the **show running-config** command is issued.*

Example 5 - Routing Specific Traffic to a Secondary ISP

In this example, two Internet connections are being used by an AOS unit. The primary Internet connection (eth 0/1, 208.61.209.253) is used to facilitate Internet access for all hosts on the private LAN (VLAN 1, 192.168.1.0 /24). The secondary Internet connection (eth 0/2, 65.162.109.201) is used strictly for email service. To prevent all the email server traffic from using the default route (which points all traffic out the primary Internet connection), the route tables must be overridden and traffic forced out the secondary Internet connection. This is accomplished with the route map **EMAIL-SERVER**, which matches all traffic coming from the server (192.168.1.2) and sends it out the secondary Internet connection. By default, the AOS firewall performs a reverse path forwarding (RPF) check to ensure spoofing attacks are blocked when traffic from the incorrect source IP address is received on a given interface. This check is based on active routes in the routing table that dictate what subnets exist off each interface. Because the routing table was overridden by a route map, the RPF check is no longer a valid method to check for spoofing attacks on the secondary Internet connection. The RPF check is disabled with the **no ip policy-class PUBLIC-SECONDARY rpf-check** command.

The firewall checks to see what ACP is applied to the outgoing interface. This information is used to determine which firewall rule to apply to the traffic. The **PRIVATE** ACP will NAT traffic to the Ethernet 0/1 address (208.61.209.253) only when it is going to be sent out an interface to which the **PUBLIC-PRIMARY** ACP is applied. The default route makes this true for all traffic except traffic sourced from the server, which is altered by the route map discussed above. Traffic altered by the route map will not match the first NAT rule in the ACP because the egress interface is changed with the **set ip next-hop** command to the secondary Internet connection (eth 0/2) where the **PUBLIC-SECONDARY** ACP is applied. Due to the fact that the egress interface ACP (**PUBLIC-SECONDARY**) does not match the ACP specified in the **nat source list** command (**PUBLIC-PRIMARY**), the first rule is not processed. Instead, the next rule in the **PRIVATE** ACP is evaluated. The traffic is matched on the second NAT rule, which checks that the traffic is routed out an interface where the **PUBLIC-SECONDARY** ACP is applied. This rule will NAT traffic to the eth 0/2 address (65.162.109.201) because the ACP applied to the egress interface matches the ACP specified in the second **nat source list** command. Administrative access is allowed on any service from the **PRIVATE** ACP because it references the **MATCHALL** ACL to self.

Only HTTPS and SSH administrative access are allowed on the **PUBLIC-PRIMARY** and **PUBLIC-SECONDARY** ACPs because they reference the **ADMIN-ACCESS** ACL. The **PUBLIC-SECONDARY** ACP contains a NAT rule that forwards all incoming email traffic from the secondary Internet connection to the email server (192.168.1.2).



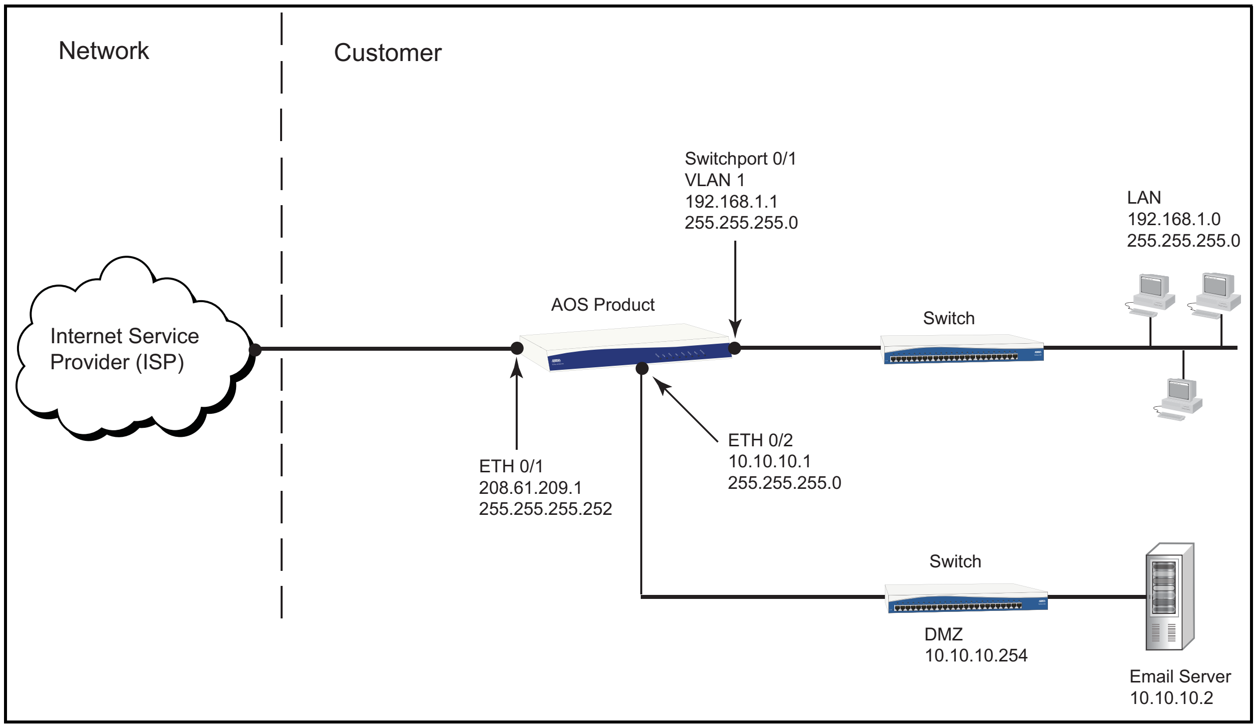
```
!
ip firewall
!
interface switchport 0/1
  no shutdown
!
interface eth 0/1
  ip address 208.61.209.253 255.255.255.252
  ip access-policy PUBLIC-PRIMARY
!
interface eth 0/2
  ip address 65.162.109.201 255.255.255.252
  ip access-policy PUBLIC-SECONDARY
!
interface vlan 1
  ip address 192.168.1.1 255.255.255.0
  ip access-policy PRIVATE
  ip policy route-map EMAIL-SERVER
  no shutdown
!
route-map EMAIL-SERVER permit 10
  match ip address EMAIL-OUT
  set ip next-hop 65.162.109.202
!
ip access-list extended ADMIN-ACCESS
  permit tcp any any eq ssh
  permit tcp any any eq https
!
ip access-list extended EMAIL-IN
  permit tcp any any eq smtp
!
ip access-list extended EMAIL-OUT
  permit ip host 192.168.1.2 any
!
ip access-list extended MATCHALL
  permit ip any any
!
```

```
ip policy-class PRIVATE
  allow list MATCHALL self
  nat source list MATCHALL interface eth 0/1 overload policy PUBLIC-PRIMARY
  nat source list MATCHALL interface eth 0/2 overload policy PUBLIC-SECONDARY
!
ip policy-class PUBLIC-PRIMARY
  allow list ADMIN-ACCESS self
!
no ip policy-class PUBLIC-SECONDARY rpf-check
ip policy-class PUBLIC-SECONDARY
  allow list ADMIN-ACCESS self
  nat destination list EMAIL-IN address 192.168.1.2
!
ip route 0.0.0.0 0.0.0.0 208.61.209.254
!
```

NOTE *A dual ISP setup like this could also be used for failover. For more information on failover, refer to the configuration guide Configuring WAN Fail-Over in AOS (available online at https://supportforums.adtran.com).*

Example 6 - Configuring a Privately Addressed DMZ

In this example, a demilitarized zone (DMZ) is created using private addresses to segregate an email server from the rest of the LAN. Public access servers are intentionally made available to traffic on the Internet to offer a service (for example, Web access, email, FTP, etc.). Servers running application services are more susceptible to application-specific exploits and should be quarantined from the LAN. Hosts in the DMZ (10.10.10.0 /24) only have the ability to initiate traffic to other hosts in the DMZ or out to the Internet. However, hosts on the LAN (192.168.1.0 /24) are able to initiate traffic to hosts in the DMZ to use the services they offer. The AOS firewall allows traffic from the DMZ back to the LAN only if it is in response to traffic sourced from the LAN. For instance, anyone on the LAN is able to access the mail server, but if the mail server is compromised by a hacker, it will not have access to anything on the LAN because the **DMZ** ACP drops the traffic to prevent it from accessing internal resources. This is accomplished with the **discard list** statement in the **DMZ** ACP. It references the **MATCHALL** ACL with the ACP **PRIVATE** keyword. This will match all traffic initiated from the DMZ and destined for the LAN where the **PRIVATE** ACP is applied. The resulting traffic will be dropped because it is a **discard list** statement. The **nat source list** command in the **DMZ** ACP facilitates Internet access for any outbound traffic from the DMZ. Inbound mail traffic is forwarded to the email server using the **nat destination list** command in the **PUBLIC** ACP. Administrative access is restricted to HTTPS and SSH on the **PUBLIC** ACP, but is open for any type of administrative traffic on the **PRIVATE** ACP.



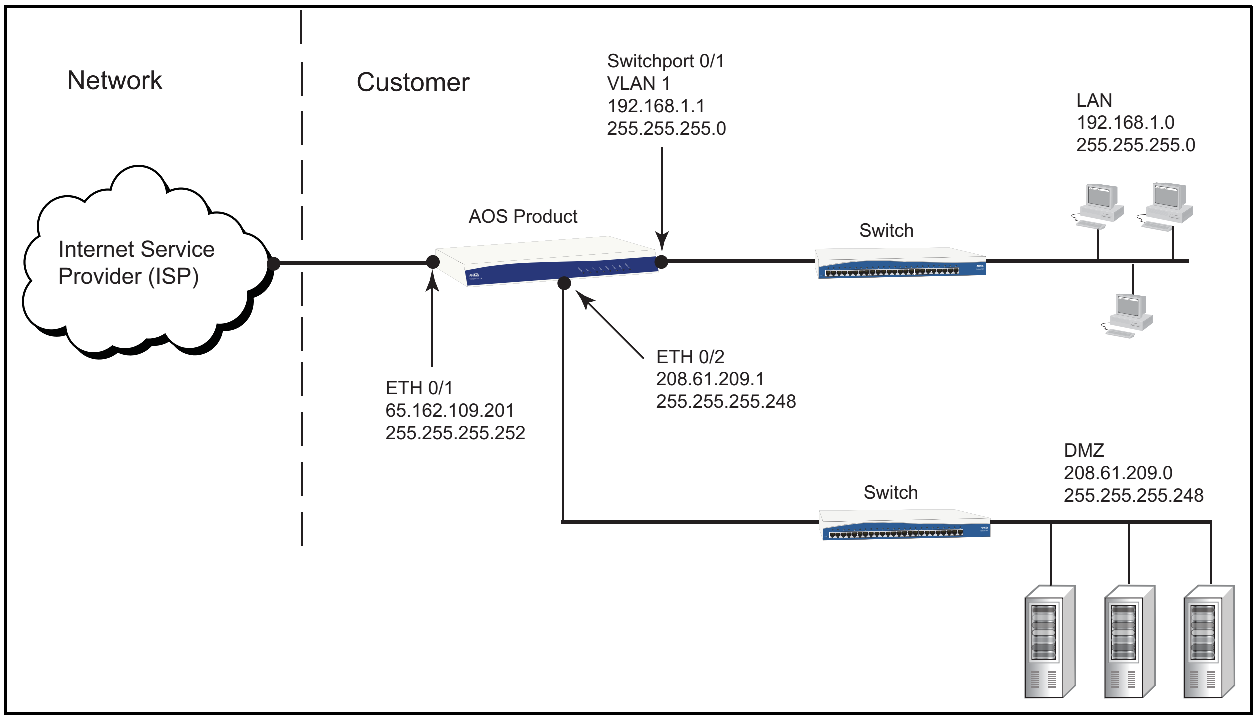
```
!
ip firewall
!
interface switchport 0/1
  no shutdown
!
interface vlan 1
  ip address  192.168.1.1  255.255.255.0
  ip access-policy PRIVATE
  no shutdown
!
interface eth 0/1
  ip address  208.61.209.1  255.255.255.252
  ip access-policy PUBLIC
  no shutdown
!
interface eth 0/2
  ip address  10.10.10.1  255.255.255.0
  ip access-policy DMZ
  no shutdown
!
ip access-list standard MATCHALL
  permit any
!
ip access-list standard WIZARD-ICS
  remark Internet Connection Sharing
  permit any
!
ip access-list extended ADMIN-ACCESS
  permit tcp any any eq ssh
  permit tcp any any eq https
!
ip access-list extended MAIL
  permit tcp any  any eq smtp
```

```
!
ip access-list extended SELF
  permit ip any  any     log
!
ip policy-class Private
  allow list SELF self
  allow list MATCHALL policy DMZ stateless
  nat source list WIZARD-ICS interface eth 0/1 overload
!
ip policy-class PUBLIC
  nat destination list MAIL address 10.10.10.2
  allow list ADMIN-ACCESS self
!
ip policy-class DMZ
  discard list MATCHALL policy PRIVATE
  nat source list WIZARD-ICS interface eth 0/1 overload
!
ip route 0.0.0.0 0.0.0.0 208.61.209.2
```

Example 7 - Configuring a Publicly Addressed DMZ

In this example, a DMZ is created using IPv4 addresses from a public block that the customer has purchased to segregate servers from the rest of the LAN. Public access servers are intentionally made available to traffic on the Internet to offer a service (i.e., Web, email, FTP, etc.). Servers running application services are more susceptible to application-specific exploits and should be quarantined from the LAN. Hosts in the DMZ (208.61.209.1 /29) only have the ability to initiate traffic to other hosts in the DMZ or out to the Internet. However, hosts on the LAN (192.168.1.0 /24) are able to initiate traffic to hosts in the DMZ to use the services they offer. The AOS firewall allows traffic from the DMZ back to the LAN only if it is in response to traffic sourced from the LAN. For instance, anyone on the LAN is able to access the servers in the DMZ, but if a server is compromised by a hacker, it will not have access to anything on the LAN because the **DMZ** ACP drops the traffic to prevent it from accessing internal resources.

This is accomplished with the **discard list** statement in the **DMZ** ACP. It references the **MATCHALL ACL** with the **policy PRIVATE** keywords. This will match all traffic initiated from the DMZ and destined for the LAN interface where the **PRIVATE** ACP is applied. The resulting traffic will be dropped because it is a **discard list** statement. The **allow list MATCHALL policy PUBLIC** statement in the **DMZ** ACP allows any outbound traffic from the DMZ to the Internet. Inbound traffic to the DMZ is allowed in using the **allow list MATCHALL policy DMZ** in the **PUBLIC** ACP. Administrative access is restricted to HTTPS and SSH on the **Public** ACP, but is open for any type of administrative traffic on the **PRIVATE** ACP.

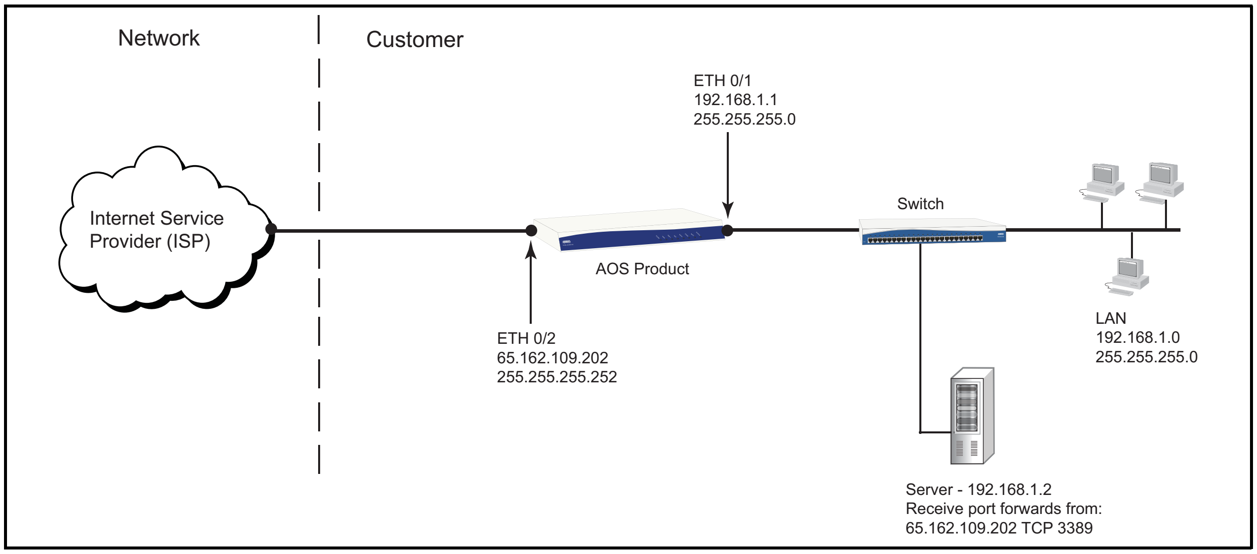


```
!
ip firewall
!
!
interface switchport 0/1
  no shutdown
!
interface vlan 1
  ip address  192.168.1.1  255.255.255.0
  ip access-policy PRIVATE
  no shutdown
!
interface eth 0/1
  ip address  65.162.109.201 255.255.255.252
  ip access-policy PUBLIC
  no shutdown
!
interface eth 0/2
  ip address  208.61.209.1  255.255.255.248
  ip access-policy DMZ
  no shutdown
!
ip access-list standard WIZARD-ICS
  remark Internet Connection Sharing
  permit any
!
ip access-list standard MATCHALL
  permit any
!
ip access-list extended ADMIN-ACCESS
  permit tcp any any eq ssh
  permit tcp any any eq https
!
ip access-list extended self
```

```
  permit ip any  any
!
ip policy-class Private
  allow list SELF self
  allow list MATCHALL policy DMZ stateless
  nat source list WIZARD-ICS interface eth 0/1 overload
!
ip policy-class PUBLIC
  allow list MATCHALL policy DMZ
 allow list ADMIN-ACCESS self
!
ip policy-class DMZ
  discard list MATCHALL policy PRIVATE
  allow list MATCHALL policy PUBLIC
!
ip route 0.0.0.0 0.0.0.0 65.162.109.202
!
```

Example 8 - Port Forward with Increased Firewall Timeout

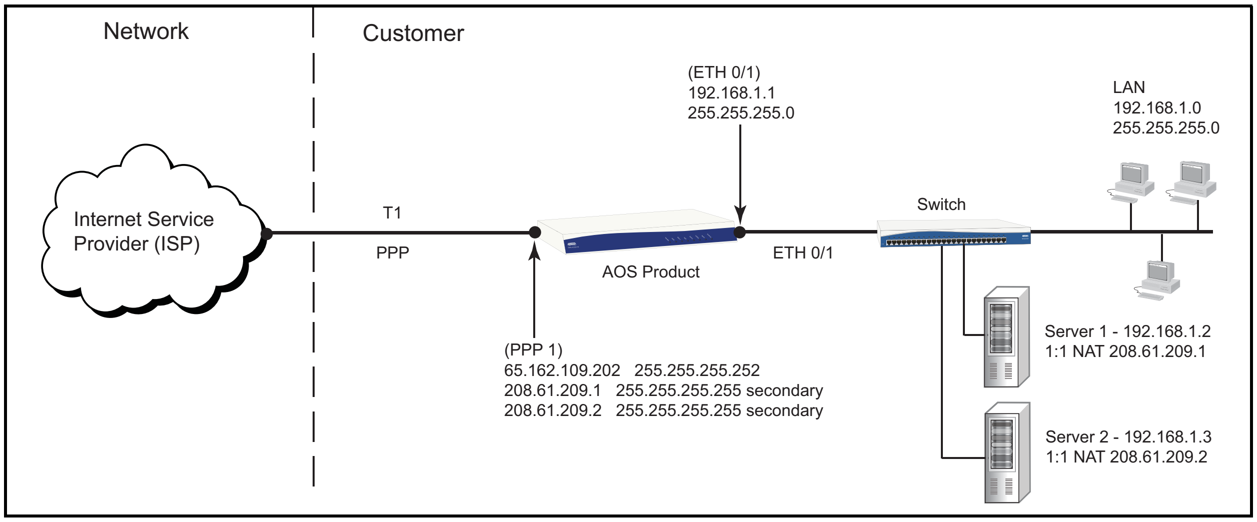
In this example a port forward is created for RDP traffic on TCP port 3389, redirecting it from the WAN interface of the AOS unit to an internal server (192.168.1.2). The firewall timeout value for this application has been increased to 8 hours (**28800** seconds) with the **ip policy-timeout** command to prevent the AOS firewall from closing out IDLE remote desktop sessions to the server.



```
!
ip policy-timeout tcp 3389 28800
!
ip firewall
!
interface eth 0/1
  ip address  192.168.1.1  255.255.255.0
  ip access-policy PRIVATE
  no shutdown
!
interface eth 0/2
  ip address  65.162.109.202  255.255.255.252
  ip access-policy PUBLIC
  no shutdown
!
ip access-list standard WIZARD-ICS
  remark Internet Connection Sharing
  permit any
!
ip access-list extended self
  remark Traffic to AOS Product
  permit ip any  any     log
!
ip access-list extended WEB-ACL-4
  remark Remote Desktop
  permit tcp any  any eq 3389
!
ip policy-class PRIVATE
  allow list SELF self
  nat source list wizard-ics interface eth 0/2 overload
!
ip policy-class PUBLIC
  nat destination list WEB-ACL-4 address 192.168.1.2
  allow list ADMIN-ACCESS self
!
ip route 0.0.0.0 0.0.0.0 65.162.109.201
!
```

Example 9 - Static 1:1 NAT

In this example, two separate static 1:1 NAT rules are created for two servers on the private side of the AOS firewall. To accomplish this, the public IPv4 addresses that are used to access the servers (208.61.209.1 and 208.61.209.2) must be applied as secondary addresses to the WAN interface (PPP 1). On the **PUBLIC** ACP, **nat destination list** commands are used to match any traffic destined to these public IPv4 addresses and direct it to the corresponding internal servers (192.168.1.2 and 192.168.1.3). In the **PRIVATE** ACP, **nat source list** commands are used to match any traffic sourced from the specified servers and change its source IPv4 address to the corresponding public IPv4 address. Any traffic not matching the first two NAT statements is caught by the catch-all NAT statement that provides Internet access to all of the other hosts on the LAN with the address 65.162.109.202. Administrative access is allowed for HTTPS and SSH by the **PUBLIC** ACP with the allow statement referencing **WEB-ACL-3**. Administrative access is allowed on all available servers from the **PRIVATE** ACP with the **allow list SELF self** command because it references an ACL that matches all traffic.



```
!
ip firewall
!
interface eth 0/1
  ip address  192.168.1.1  255.255.255.0
  ip access-policy PRIVATE
  no shutdown
!
interface t1 1/1
  clock source line
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface ppp 1
  ip address  65.162.109.202  255.255.255.252
  ip address  208.61.209.1  255.255.255.255 secondary
  ip address  208.61.209.2  255.255.255.255 secondary
  ip access-policy PUBLIC
  no shutdown
  cross-connect 1 t1 1/1 1 ppp 1
!
ip access-list standard WIZARD-ICS
  remark Internet Connection Sharing
  permit any
!
ip access-list extended self
  remark Traffic to NetVanta
  permit ip any  any     log
!
ip access-list extended WEB-ACL-3
  permit tcp any  any eq https
  permit tcp any  any eq ssh
!
ip access-list extended WEB-ACL-4
  remark 1:1 NAT 208.61.209.1 > 192.168.1.2
  permit ip any  host 208.61.209.1
!
ip access-list extended WEB-ACL-5
  remark 1:1 NAT 192.168.1.2 > 208.61.209.1
  permit ip host 192.168.1.2  any
!
ip access-list extended WEB-ACL-6
  remark 1:1 NAT 208.61.209.2 > 192.168.1.3
  permit ip any  host 208.61.209.2
!
ip access-list extended WEB-ACL-7
  remark 1:1 NAT 192.168.1.3 > 208.61.209.2
  permit ip host 192.168.1.3  any
!
ip policy-class PRIVATE
  allow list SELF self
  nat source list WEB-ACL-5 address 208.61.209.1 overload
  nat source list WEB-ACL-7 address 208.61.209.2 overload
  nat source list WIZARD-ICS interface ppp 1 overload
!
ip policy-class PUBLIC
  nat destination list WEB-ACL-4 address 192.168.1.2
  nat destination list WEB-ACL-6 address 192.168.1.3
  allow list WEB-ACL-3 self
!
ip route 0.0.0.0 0.0.0.0 ppp 1
!
```

NOTE *The command **clock source line** is enabled by default. Therefore, this command will not appear in the output when the **show running-config** command is issued.*

Command Summary Table

The following table summarizes the minimum steps required to configure IPv4 Firewall on an AOS product.

Table 6. Firewall Configuration Steps

Step	Command	Description
Step 1	Boot up the unit, then telnet to the unit (**telnet** *<ipv4 address>*)	Access the CLI.
Step 2	(config)#**ip firewall**	Enable the IPv4 firewall functionality in AOS.
Step 3	(config)#**ip access-list [extended | standard]** *<ipv4 acl name>*	Create a standard or extended ACL and enter the IPv4 ACL configuration mode.
	(config-std-nacl)#**[permit | deny]** *<source>* **[log] [track** *<name>***]**	If you created a standard ACL, specify the packet source IP information and decide whether the ACL will permit or deny traffic based on a match. The optional **track** *<name>* parameter associates the ACL entry with a particular track. The optional **log** parameter specifies that any entries that match the ACL criteria will be logged for **show** and **debug** use. Refer to *Define Permissions on page 19* for more information.
	(config-ext-nacl)#**[permit | deny]** *<protocol> <source> <source port> <destination> <destination port>* **[log] [track** *<name>***] [fragments]**	If you created an extended ACL, specify whether the ACL will **permit** or **deny** traffic based on protocol, source IP information, or destination IP information. The optional **track** *<name>* parameter associates the ACL entry with a particular track. The optional **log** parameter specifies that any entries that match the ACL criteria will be logged for **show** and **debug** use. The optional **fragments** parameter specifies that the ACL entry is only matched by non-initial fragments. Refer to *Define Permissions on page 19* for more information.
(Optional)	(config-std-nacl)#**remark** *<remark> or* (config-ext-nacl)#**remark** *<remark>*	Associate a comment with the entry to further describe the purpose of the standard or extended ACL.

Table 6. Firewall Configuration Steps

Step	Command	Description
Step 4	(config)#**ip policy-class** *<ipv4 acp name>*	Create an IPv4 ACP and enter the ACP configuration mode.
Allow Traffic Based on ACL Entries	(config-policy-class)#**allow list** *<ipv4 acl name>* **[self | policy** *<ipv4 acp name>***] [stateless]**	Specify an ACL to determine which IP packets are allowed to enter the interface to which the IPv4 ACP is assigned, and create a policy session in the firewall.
	(config-policy-class)#**allow reverse list** *<ipv4 acl name>* **[self | policy** *<ipv4 acp name>***] [stateless]**	Specify an ACL to determine which IP packets are allowed to enter the interface to which the IPv4 ACP is assigned, and create a policy session in the firewall. The **reverse** keyword instructs the firewall to use the source information as the destination information and vice versa when attempting matches against the specified ACL.
Discard Traffic Based on ACL Entries	(config-policy-class)#**discard list** *<ipv4 acl name>* **[self | policy** *<ipv4 acp name>***]**	Specify an ACL to determine which packets are discarded after being received on the interface to which the IPv4 ACP is assigned.
Apply NAT to the Dest. IPv4 Address	(config-policy-class)#**nat destination list** *<ipv4 acl name>* **[address** *<ipv4 ip address>* **| vrf** *<name>* **| port** *<port number>***] [no-alg]**	Translate the destination IPv4 address to a specified IPv4 address, and create a policy session.
Step 5	(config)#**interface** *<interface>* (config-interface)#**ip access-policy** *<ipv4 acp name>*	Enter the appropriate interface configuration mode, and apply the IPv4 ACP to the interface.

Troubleshooting

There are a number of methods available to assist you in troubleshooting the IPv4 firewall configuration on your unit. The following list provides methods available through the CLI.

- **Show** commands provide a means to verify your configuration (refer to *Using Show Commands* below).
- **Clear** commands clear statistics for the associated feature or clear traffic flows (refer to *Using Clear Commands on page 80*).
- **Debug** commands display events as they occur to help you better interpret possible problems (refer to *Enabling Debug Commands on page 81*).
- The event log thresholds and messages are important features of AOS to understand and use (refer to *Maintaining Log Thresholds on page 83* and *Managing Event Messages on page 84*).

The Security Dashboard is also available to assist with troubleshooting through the GUI. Refer to the configuration guide *Configuring the Security Dashboard in AOS* (available online at https://supportforums.adtran.com).

Using Show Commands

After configuring the IPv4 firewall, several **show** commands can be issued in the CLI to assist in troubleshooting and verifying your configuration. Enter **show** commands from the Enable prompt or use the **do show** command from any configuration prompt. The following table lists these commands and their descriptions. Refer to *Show Commands Sample Output on page 76* for examples.

Table 7. Show Commands

Command	Description
show ip access-lists [<*ipv4 acl name*>**]**	Displays currently configured ACLs. Specifies a particular IPv4 ACL to display using the optional <*ipv4 acl name*> parameter.
show ip ffe	Displays RapidRoute entries. There are multiple variations of this command. Refer to the *AOS Command Reference Guide* (available online at https://supportforums.adtran.com) for a detailed description of this command.
show ip policy-class [<*ipv4 acp name*> **| host-sessions]**	Displays currently configured ACPs. Specifying an IPv4 ACP using the <*ipv4 acp name*> parameter displays information only for the named IPv4 ACP. Using the **host-sessions** keyword displays a session count for each unique host IPv4 address (if **max-host-sessions** is configured for the ACP).
show ip policy-sessions [<*ipv4 acp name*> **| any-vrf | vrf** <*name*>**] [include-deleted | timeline]**	Displays a list of all policy sessions that are active in the unit. Each entry is displayed in two lines of information. The first line shows the protocol with the TTL for the session and the destination ACP. The second line shows the source address and port, the destination address and port, and optionally, the NAT IPv4 address and port. Specifying a particular IPv4 ACP using the <*ipv4 acp name*> parameter limits the output to only the named IPv4 ACP. If a VRF is specified, the output displays a list of current policy sessions for the specified VRF instance. If no VRF is given, output for the default (unnamed) VRF is displayed. Using **any-vrf** will display all policy sessions for all VRFs. Policy sessions marked for deletion whose deletion is still pending can be viewed by using the optional **include-deleted** keyword (active policy sessions will also be displayed). A timeline of policy session creations and peak numbers of sessions over the last 24 hours can also be displayed using the **timeline** parameter.
show ip policy-stats [<*ipv4 acp name*>**]**	Displays current ACP statistics. To limit the output to display statistics for a specific IPv4 ACP, use the optional <*ipv4 acp name*> parameter.
show ip traffic [netstat | realtime]	Displays IPv4 traffic statistics. The optional **netstat** parameter displays active TCP connections. The optional **realtime** parameter displays the output continuously as it occurs.
show running-config ip access-list	Displays the current running configuration for all configured ACLs.
show running-config ip policy-class	Displays the current running configuration for all configured ACPs.

Show Commands Sample Output

The following sample output are provided to enhance your understanding of the information provided for each **show** command.

```
#show ip access-lists
* - Indicates access list entry disabled by track.
Standard IP access list MATCHALL
  permit 192.168.1.0 0.0.0.255 (31337 matches)
Standard IP access list SERVER1_OUT
  permit host 192.168.1.100 (0 matches)
Extended IP access list CORPORATE_TRAFFIC
  permit ip 192.168.1.0 0.0.0.255 192.168.3.0 0.0.0.255 (432829 matches)
```

```
Extended IP access list CORPORATE_TRAFFIC_IN
  permit ip 192.168.3.0 0.0.0.255 192.168.1.0 0.0.0.255 (2194 matches)
Extended IP access list REMOTE_USER_TRAFFIC
  permit ip 192.168.1.0 0.0.0.255 10.10.10.0 0.0.0.255 (178 matches)
Extended IP access list REMOTE_USER_TRAFFIC_IN
  permit ip 10.10.10.0 0.0.0.255 192.168.1.0 0.0.0.255 (11 matches)

#show ip ffe summary
Ingress     MaxEntries   Entries          Hits            Misses         Drops
----------  ----------   ----------       ----------      ----------     ----------
eth 0/1     4096         78               22294074        219826172      0
eth 0/2     4094         117              89456500        262988307      1021
----------  ----------   ----------       ----------      ----------     ----------
Global      16384        195              111750574       482814479      1021

#show ip policy-class
3 current sessions (100 peak of 30000 max)

Policy-class "TRUSTED":
  1 current sessions (10000 max)
  Discards/Allows/NAT: 0/54/3224
  Entry 1 - allow list CORPORATE_TRAFFIC
  Entry 2 - allow list REMOTE_USER_TRAFFIC
  Entry 3 - nat source list SERVER1_OUT address 208.61.209.1 overload
  Entry 4 - nat source list MATCHALL address 208.61.209.10 overload

Policy-class "UNTRUSTED":
  2 current sessions (10000 max)
  Discards/Allows/NAT: 0/54/0
  Entry 1 - allow list CORPORATE_TRAFFIC_IN
  Entry 2 - allow list REMOTE_USER_TRAFFIC_IN

#show ip policy-class host-sessions

Policy-class “PRIVATE”:
  100 policy-sessions allowed per source address.
  Src IP Address                              Sessions
  ---------------                             --------
  192.168.1.100                               1
  192.168.1.101                               35
  192.168.1.121                               100 (maximum allowed)

Policy-class “PUBLIC”:
  No limit for policy-sessions allowed per host.
```

NOTE *The output for the* ***show ip policy sessions*** *command can be rather lengthy. Only the first screen of output is shown to provide an example of the information this command displays.*

```
#show ip policy-sessions
Src Vrf (if not default), Src policy class:
Protocol (TTL) [in crypto map] -> [out crypto map] Dest VRF, Dest policy-class
Src IP Address  Src Port  Dest IP Address  Dst Port NAT IP Address   NAT Port
--------------- --------  ---------------  -------- ---------------  --------
Policy class "PRIVATE":
tcp (579) -> PUBLIC
10.10.20.9      3481      67.195.187.192   5050      s 208.61.209.1  3481
tcp (586) -> PUBLIC
10.10.20.9      3490      98.137.130.66    443       s 208.61.209.1  3490
tcp (28800) -> SELF
10.10.20.9      2176      192.168.0.25     23
udp (60) -> PUBLIC
10.10.20.9      2233      208.61.208.253   2233      s 208.61.209.1  1057
udp (39) -> PUBLIC
192.168.2.4     4500      66.0.238.250     4500      s 208.61.209.1  4500
udp (39) -> PUBLIC
192.168.2.4     4500      208.61.208.235   4500      s 208.61.209.1  1061
tcp (544) -> PUBLIC
192.168.5.3     61064     69.63.180.44     80        s 208.61.209.1  61064
Policy class "PUBLIC":
tcp (600) -> PRIVATE
67.58.88.6      1027      208.61.209.1     47624     d 192.168.5.5   47624
Policy class "SELF":
icmp (46) -> PRIVATE
192.168.5.1 2             192.168.5.5      2
icmp (46) -> PRIVATE
192.168.5.1 1             192.168.5.61     1

Policy class "DEFAULT":

#show ip policy-stats
3 current sessions (100 peak of 30000 max)

Discards/Allows/NAT: 0/145/543

Policy-class "TRUSTED":
  1 current sessions (10000 max)
  Discards/Allows/NAT: 0/54/3224
  Entry 1 - allow list CORPORATE_TRAFFIC
    10211717 in bytes, 1184 out bytes, 1140 hits
  Entry 2 - allow list REMOTE_USER_TRAFFIC
    0 in bytes, 0 out bytes, 0 hits
  Entry 3 - nat source list SERVER1_OUT
    address 141.158.56.58 overload 0 in bytes, 0 out bytes, 0 hits
  Entry 4 - nat source list MATCHALL
    address 141.158.56.62 overload 66422200 in bytes, 230583087 out bytes, 31332 hits

Policy-class "UNTRUSTED":
  2 current sessions (10000 max)
  Entry 1 - allow list CORPORATE_TRAFFIC_IN
    1306324 in bytes, 139295 out bytes, 2194 hits
  Entry 2 - allow list REMOTE_USER_TRAFFIC_IN
    1051 in bytes, 128 out bytes, 11 hits
```

```
#show ip traffic
IP statistics:
Routing discards: 0
Rcvd: 15873 total, 7617 delivered
0 header errors, 0 address errors
0 unknown protocol, 0 discards
0 checksum errors, 0 bad hop counts
Sent: 8281 generated, 4459 forwarded
0 no routes, 0 discards
Frags: 0 reassemble required, 0 reassembled, 0 couldn't reassemble
0 created, 0 fragmented, 0 couldn't fragment

UDP statistics:
Rcvd: 3822 total, 0 checksum errors, 0 no port
Sent: 3822 total

TCP statistics:
Retrans Timeout Algorithm: 0
Min retrans timeout (ms): 0
Max retrans timeout (ms): 0
Max TCP Connections: 0
0 active opens, 64 passive opens, 0 failed attempts
5 establish resets, 1 establish current
3795 segments received, 4459 segments sent, 26 segments retransmitted

#show running-config ip access-list
ip access-list standard MATCHALL
  permit any
!
ip access-list standard TEST
  permit host 10.23.166.13
  permit host 10.22.120.12
  permit any
!
ip access-list extended EXAMPLE
  permit ip any  host 10.22.130.6
!
ip access-list extended IT_CORP
  permit ip any  172.16.0.0 0.15.255.255
  deny   ip any  10.22.254.0 0.0.0.255
  deny   ip any  host 10.100.23.10
  deny   ip any  host 10.100.23.11
  deny   ip any  host 10.100.23.12
  deny   ip any  host 10.100.23.1
  deny   ip any  host 10.100.43.250
  permit ip any  10.0.0.0 0.255.255.255
!
ip access-list extended EXAMPLE
  permit ip any  hostname example.org
  permit ip any  hostname www.example.org
  permit ip any  hostname www.example.com
  permit ip any  hostname otherexample.com
  permit ip any  hostname www.otherexample.com
!
ip access-list extended WEB
  deny   tcp host 10.22.110.33  any eq www
  permit tcp any  any eq 8080
  permit tcp any  any eq https
  permit tcp any  any eq www
!
end
```

```
#show running-config ip policy-class
no ip policy-class FREEPASS rpf-check
ip policy-class FREEPASS
  discard list KEEPOUT
  allow list MATCHALL stateless
!
ip policy-class EXAMPLE
  allow list MATCHALL
!
ip policy-class OFFICEVLAN
  allow list MATCHALL stateless
!
no ip policy-class PRIVATE rpf-check
ip policy-class PRIVATE
  allow list EXAMPLE
  discard list BADIP
  nat source list WEB interface eth 0/1 overload
  allow list MATCHALL stateless
!
no ip policy-class PROTECTIT rpf-check
ip policy-class PROTECTIT
  discard list IT_CORP
  allow list MATCHALL stateless
!
!
end
Using Clear Commands
```

Clear commands are issued from the Enable mode prompt and clear all statistics associated with the specified feature.

Table 8. Clear Command Summary

Command	Description
clear ip access-list [<*ipv4 acl name*>**]**	Clears all counters associated with all ACLs or a specified ACL. Using the optional <*ipv4 acl name*> parameter clears the statistics only for the specified IPv4 ACL, rather than all configured ACLs.
clear ip policy-sessions vrf <*name*> <*ipv4 acp name*> **[ahp | esp | gre | icmp | tcp | udp |** <*protocol*>**]** <*source ipv4 address*> <*source port*> <*destination ipv4 address*> <*dest port*> **[destination | source]** <*ipv4 nat*> <*nat port*>	Clears all IP policy sessions or a specified session. Delete specific policy sessions by using the optional parameters that follow **clear ip policy-sessions**. Clear per VRF using **vrf** <*name*>. Clear per IPv4 ACP using <*ipv4 acp name*>. Some ALGs (like SIP) will not allow policy sessions created by the ALG to be manually deleted. The second half of this command, beginning with the source IPv4 address, can be copied and pasted from a row in the **show ip policy-sessions** table for easier use.
clear ip policy-stats [<*ipv4 acp name*>**] [entry** <*number*>**]**	Clears the hit counts on individual ACP entries. Without an argument, the command clears the ACP statistics for all entries on all IPv4 ACPs. With the <*ipv4 acp name*> argument only, the command clears the hit ACP statistics for all entries on the specified IPv4 ACP only. This command is useful to verify that the correct IPv4 ACP entries are being matched.

Enabling Debug Commands

Another method for troubleshooting the IPv4 firewall configuration is enabling and viewing **Debug** commands. These commands are issued from the Enable mode prompt, and are displayed (real time) to the terminal (or Telnet) screen.

Table 9. Debug Command Summary

Command	Description
debug ip firewall [**vrf** <*name*>]	Displays session information when IPv4 firewall sessions are created and deleted, as well as information-level policy log messages. You must change the threshold setting (**ip firewall policy-log threshold 1**) to see all policy log messages. The optional **vrf** keyword displays information for the specified VRF instance.
debug ip tcp [**events**][**md5**]	Activates debug messages associated with TCP state changes, session allocation and deallocation, and packet information (for example, sequence numbers, acknowledgment numbers, and packet length) in AOS. These debug events are logged for packets that are sent or received from the router. Forwarded TCP packets are not included. The optional **events** keyword activates only messages regarding TCP state changes and TCP session allocation. The optional **md5** keyword displays TCP MD5 authentication data.
debug ip udp	Activate debug messages associated with UDP send and receive events in AOS. The message **no listener** means that there is no service listening on this UDP port (for example, the data is discarded).

NOTE *Turning on a large amount of debug information can adversely affect the performance of your unit.*

Debug Commands Sample Output and Explanation

```
#debug ip firewall
2009.09.15 09:33:19 FIREWALL Deleting Association
2009.09.15 09:33:19 FIREWALL   Assoc Index = 6242787, Count (total, policy-class) = 127, 126
2009.09.15 09:33:19 FIREWALL   nat source -> 10.22.19.48, flags = 0x20000026, 0x00000004, timeout = 4
2009.09.15 09:33:19 FIREWALL   Selector1: Dir=PRIVATE, int=eth 0/2, Protocol=6  cookie-> eth 0/1
2009.09.15 09:33:19 FIREWALL     SrcIp: 192.168.48.1, DstIp: 10.100.50.2
2009.09.15 09:33:19 FIREWALL     SrcPort: 41801, DstPort: 80
2009.09.15 09:33:19 FIREWALL   Selector2: Dir=PUBLIC, int=eth 0/1, Protocol=6  cookie-> eth 0/2
2009.09.15 09:33:19 FIREWALL     SrcIp: 10.100.50.2, DstIp: 10.22.19.48
2009.09.15 09:33:19 FIREWALL     SrcPort: 80, DstPort: 1034
```

In the output above, the first line indicates that an IPv4 firewall policy session is being deleted. This means that the opening created to allow matching traffic through the firewall has been closed. If the ACP configuration allows such traffic, traffic may create a new policy session in the future.

In line 2, **Assoc Index** is a unique global identifier of this specific policy session. **Count** indicates the current total number of firewall policy sessions across the unit, and the total number of firewall policy sessions for the ACP (specified in **Selector1**).

In line 4, **Selector1** describes the initial traffic flow that opened this IPv4 firewall policy session. This traffic flow was allowed by the firewall ACP configuration or created by an ALG.

In line 7, **Selector2** describes the return traffic flow corresponding to **Selector1** to allow bidirectional traffic. If NAT is not involved, the source and destination IPv4 addresses and ports will mirror one another on **Selector1** and **Selector2**. If NAT is involved, the source or destination information might be translated, depending on whether source NAT or destination NAT is used.

In both lines 4 and 7, **Dir** identifies the ACP used by traffic matching this selector. For **Selector1**, this is the ACP applied to the ingress interface(s) for initial traffic. For **Selector2**, this is the ACP applied to the interface(s) on which return traffic arrives. Compare to **int**, the interface on which initial or response traffic arrived (depending on the selector). **Protocol** is the IANA assigned protocol number for the traffic, for example, TCP (6), UDP (17), etc. (refer to http://www.iana.org/assignments/protocol-numbers/). **Cookie** describes the destination interface(s) of traffic matching this selector. If load sharing is used, multiple interfaces can appear. If PBR is used, the interface can differ from the route table based on the routing policy applied.

Other similar messages could appear when:

- A policy session is created (for example, a new traffic flow is allowed through the firewall).
- A pending policy session (created by an ALG) becomes active (for example, when expected traffic arrives to be allowed through the firewall).
- A policy session is reworked due to a route table change or Address Resolution Protocol (ARP) cache change.

NOTE *If traffic is being blocked by the IPv4 firewall, you can use the* ***debug ip firewall*** *output to determine when the firewall is allowing and blocking traffic. You can then examine your ACP configuration and your policy timeout configuration to further troubleshoot the issue.*

```
#debug ip tcp events
09:55:56:TCP 0: Allocating session ----------------
09:55:56:TCB 3204:state change: FREE->SYNRCVD
09:55:56:TCB 3204:new connection from 10.22.1.123:2081 to 10.22.190.9:23
09:55:56:TCB 3204:state change: SYNRCVD->ESTABLISHED [10.22.1.123:2081]
TCB 3204:Preparing to close[1][443d2790] 2 bytes left to TX
TCB 3204:Preparing to close[2][443d2790] 0 bytes left to TX
09:56:08:TCB 3204:state change: ESTABLISHED->FINWAIT1 [10.22.1.123:2081]
09:56:08:TCB 3204:state change: FINWAIT1->FINWAIT2 [10.22.1.123:2081]
TCB 3204:De-allocating session
09:56:08:TCB 3204:De-allocating session ----------------
```

NOTE *In the* ***debug ip tcp events*** *information, TCB stands for TCP task control block. The numbers that sometimes appear next to TCB (for example,* ***TCB 3204*** *in the previous example) represent the TCP session number. This allows you to differentiate debug messages for multiple TCP sessions.*

In the output above, the unit is listening on certain TCP ports based on which services are configured (such as, Telnet or FTP server). In the first line, **Allocating session** means it has received a TCP SYN packet on a port to which it is listening and is now allocating one of its available TCP sessions.

In line 2, the TCP state changes from a **FREE** state to a **SYNRCVD** state since the unit received the TCP SYN packet.

In line 3, the TCP SYN packet was received from IPv4 address 10.22.1.123, port 2081 and is destined to IPv4 address 10.22.190.9, port 23 (which is Telnet).

In line 4, the TCP state transitions from **SYNRCVD** to the **ESTABLISHED** state. This represents an open connection. At this point, the two devices should be able to communicate.

In line 5, **Preparing to close [1][443d2790] 2 bytes left to TX**, the unit receives a request from the remote device to terminate the connection. However, there are still two bytes left to transmit to finish the previous transaction.

In line 6, **Preparing to close [2][443d2790] 0 bytes left to TX**, the transaction is complete and the connection is ready to be terminated.

In line 7, **ESTABLISHED->FINWAIT1 [10.22.1.123:2081]**, the TCP state transitions from **ESTABLISHED** to the **FINWAIT1** state to start the process of closing the connection after receiving the request from the remote to terminate the connection.

In line 8, **FINWAIT1->FINWAIT2 [10.22.1.123:2081]**, the TCP state transitions from the **FINWAIT1** to the **FINWAIT2** state, which means that the unit agrees to terminate the connection.

```
#debug ip udp
13:12:26: TX: src=10.22.130.253:1040, dst=10.100.23.1:69, 26 bytes,
13:12:26: RX: src=10.100.23.1:1817, dst=10.22.130.253:1040, 12 bytes,
13:12:26: TX: src=10.22.130.253:1040, dst=10.100.23.1:1817, 16 bytes,
13:12:26: RX: src=10.100.23.1:1817, dst=10.22.130.253:1040, 12 bytes,
```

The output above indicates the direction of each packet (**TX** or **RX**), the source and destination IPv4 addresses, the source and destination port numbers, and the number of bytes transferred. In this example, the destination port **69** in the first packet indicates that it is a TFTP transfer.

Maintaining Log Thresholds

Logs can be examined for information to assist you in troubleshooting or in determining the kinds of attacks that have targeted your system. You can also view events as they occur on the console by activating the **debug** commands from the Enable mode as described in *Enabling Debug Commands on page 81*.

The AOS logs attack and ACP events to the console. The first attack event and the first ACP event are displayed, but no other events are displayed again until after a certain number of events have occurred. This number is determined by the settings configured using the **ip firewall attack-log threshold** *<value>* and **ip firewall policy-log threshold** *<value>* commands. For example, assume that the attack log threshold is 70. The first attack event will be displayed, and once a total of 70 more attack events occur, the 71st attack event will be displayed.

The default value for each of these commands is **100**, but when troubleshooting dropped packets, the thresholds should be set to **1**. By setting the thresholds to **1**, every dropped packet will display, making it easier to determine the exact circumstances causing the packet to be dropped. Some ACP log messages have a severity level that will prevent them from being displayed unless the appropriate **debug** command has been enabled. Refer to *Managing Event Messages on page 84* for more information.

Managing Event Messages

There are multiple priority levels for event messages. You can manage these messages in several ways, based on their assigned priority level. The levels are listed below, from least critical to most critical.

Priority Level Number	Priority Level
5	Debug
4	Information
3	Notice
2	Warning
1	Error
0	Fatal

There are two management options for the event messages displayed on the console. The default behavior displays levels 0 to 3 (for example, Fatal, Error, Warning, and Notice messages). To display events for all priority levels, issue the command **debug ip firewall**. Issue the **no debug ip firewall** command to stop displaying the events.

There are additional management options available for event history storage, email notification, and syslog forwarding. If event history storage is enabled (using the **event-history on** command), by default AOS logs all messages with priority levels 0 through 4 (for example, Fatal, Error, Warning, Notice, and Information messages). Email notification (enabled using the **logging email on** command) also logs priority levels 0 through 4 by default, but syslog forwarding (enabled using the **logging forwarding on** command) logs priority levels 0 through 3 by default. You can use the following commands to change the default behavior and set an explicit priority level.

- **event-history priority** <*priority level*> sets the threshold for events stored in the event history. The event log is displayed using the **show event-history** command.
- **logging email priority-level** <*priority level*> sets the threshold for events sent to the configured email addresses (specified using the **logging email address-list** command).
- **logging forwarding priority-level** <*priority level*> sets the threshold for events sent to the configured syslog server (specified using the **logging forwarding receiver-ip** command).

NOTE *When setting the* ***<priority level>****, keep in mind that your selection includes not only the specified priority level but those below it, as well. For example, when priority 3 is selected, events with priority 3, 2, 1, and 0 are logged.*

For more information on logging, refer to the guides *Configuring Console Logging in AOS* and *Configuring Email Logging in AOS* (available online at https://supportforums.adtran.com).

Table 10. IPv4 Firewall Events

Event Message	Priority Level
All attack-log messages (Refer to *Appendix A. Attack Log Messages on page 87*)	Error (1)
Service access request successful	Information (4)
No Access Policy matched, dropping packet	Information (4)
Deny Access Policy matched, dropping packet	Information (4)
Connection terminated. Bytes transferred: <*value*>	Information (4)
Connection closed. Bytes transferred: <*value*>	Information (4)
Connection timed out. Bytes transferred: <*value*>	Information (4)
Maximum number of global associations reached, dropping packet	Notice (3)
Maximum number of associations reached on <*policy-class name*> policy-class, dropping packet	Notice (3)
Available heap free does not allow for a new association, dropping packet	Notice (3)
Unable to initialize Association Protocol Info, deleting packet	Notice (3)
Unable to send SYN packet	Notice (3)
Send final ACK to target failed	Notice (3)
Unable to get Port for Protocol <*protocol number*>	Notice (3)
ADFreeNatPort: Unable to get PortMap for NAT <*ip address:port*> Proto <*protocol number*> vrf <*name*>	Notice (3)
Unable to free Unknown Protocol NAT port for <*ip address:port*> vrf <*name*>	Notice (3)
Unable to free TCP NAT port for <*ipv4 address:port*> vrf <*name*>	Notice (3)
Unable to free UDP NAT port for <*ipv4 address:port*> vrf <*name*>	Notice (3)
Unable to free ICMP NAT port for <*ipv4 address:port*> vrf <*name*>	Notice (3)
Unable to free GRE NAT port for <*ip v4address:port*> vrf <*name*>	Notice (3)
Unable to free Unknown Protocol NAT port for <*ipv4 address:port*> vrf <*name*>	Notice (3)
Attempt to de-register port map for unavailable NIP <*ipv4 address*><*ipv4 address*>	Notice (3)
ADLAddNatPort: Unable to get PortMap for NAT <*ipv4 address*> Proto <*protocol number*>	Notice (3)
ADLAddNatPort: Trying to add reference to unused port <*port*> for NAT <*ipv4 address*> Proto <*protocol number*>	Notice (3)
ADLAddNatPort: Trying to add reference to unreferenced port <*port*> for NAT <*ipv4 address*> Proto <*protocol number*>	Notice (3)
ADLDelNatPort: Port to free was not found for <*ipv4 address:port*>, vrf <*vrf id*>	Notice (3)
IGWGetPortByAlgId: Unable to get PortMap for NAT <*ipv4 address*> Proto <*protocol number*>	Notice (3)
IGWGetPortPairByAlgId: Unable to get PortMap for NAT <*ipv4 address*> Proto <*protocol number*>	Notice (3)
ADAlgRegisterNatPorts: Invalid Range StartPort <*port*> EndPort <*port*>	Notice (3)
ADAlgRegisterNatPorts: Trying to register twice. AlgId <*ALG id*> Protocol <*protocol number*>	Notice (3)
ADAlgRegisterNatPorts: Some ports in the specified Range already Registered AlgId <*ALG id*> Protocol <*protocol number*> StartPort <*port*> EndPort <*port*>	Notice (3)

Table 10. IPv4 Firewall Events *(Continued)*

NAT port pool helper cannot find pool for <*ipv4 address*> (vrf <*name*>) for deferred deletion	Notice (3)
Invalid FTP PASV cmd reply seen, dropping packet	Notice (3)
FTP Get port failed	Notice (3)
FTP PASV cmd response came without request, dropping packet	Notice (3)
H.323: Unable to get Nat port	Notice (3)
H.323: Failed to make H323_H245 Connection	Notice (3)
H.323: Failed to Allocate Nat Port	Notice (3)
H.323: Failed to make connection for H323T120	Notice (3)
H.323: Failed to make connection for H323RtpRtcp	Notice (3)
IRC: No of Messages are more than MAX_IRC_REQUESTS	Notice (3)
IRC: Size of Message is more than MAX_IRCSIZE	Notice (3)
IRC: Unable to create dynamic association for IRC	Notice (3)
Stored RPC transaction Id doesn't match server response, dropping packet	Notice (3)
RPC Server's response is undecipherable, dropping packet	Notice (3)
RTSP: Failed to Nat Port for RTSP connection	Notice (3)
RTSP: Failed to Create RTSP Data connection	Notice (3)
Not creating passive association with NAT port of zero	Notice (3)
ALGSipInit: Failed to set app process	Notice (3)
ALGSipProcess: Received non-SIP packet	Notice (3)
ALGSipProcess: Received unsupported SIP request (<*request*>)	Notice (3)
ALGSipProcess: ALGSipMangleMessage returned error	Notice (3)
ALGSipProcess: Failed to create association for RTP	Notice (3)
Unable to find Call-ID in message	Notice (3)
ALGSipMangleMessage: Unable to NAT Request-Line	Notice (3)
SIP ALG: Failed to add contact NAT port to RegPasv table	Notice (3)
ALGSipMangleMessage: Failed to add Undo information to NAT table	Notice (3)
ALGSipMangleMessage: Failed to add Call-Id Undo information to NAT table	Notice (3)
ALGSipMangleMessage: Failed to create association for RTP	Notice (3)
ALGSipMangleMessage: Failed to create association for RTCP	Notice (3)
ALGSipNewConnection: Failed to create association	Notice (3)
Security policy unavailable for policy-class, dropping packet	Warning (2)
IN bound Access Policy not found, dropping packet	Warning (2)

Appendix A. Attack Log Messages

This appendix provides a list of all the possible attack log messages that can appear on an AOS unit. Threats can possibly be attacks, but not necessarily as they could also be caused by misconfigurations or peculiarities in the network. Threats have been categorized and been assigned a weight based on their possible severity. Threats with a higher severity have the potential to be more compromising to hosts behind the firewall than threats with a lower severity.

NOTE *Keep in mind that the IPv4 firewall in your AOS device provides protection against all of these potential threats.*

Each message indicates that an event has occurred, which has the potential to pose a threat to host(s) behind the IPv4 firewall. In all cases enumerated below, the message indicates that the firewall has detected the threat and has protected the hosts behind the firewall from the attack. Each threat listed includes the actual message seen in the event history, a description of the event and its cause (or possible causes), and the action AOS takes in response to the threat. It also includes the ID, short definition, category, and weight of the threat. Except for the category (which is visible from neither the GUI nor the CLI) and the message itself (which appears in the event history) all of this information can be viewed in the GUI's Security Dashboard. The only items that are visible from the CLI are the ID, short definition, and weight.

For more information on accessing and using the Security Dashboard feature, refer to the configuration guide *Configuring the Security Dashboard in AOS* (available online at https://supportforums.adtran.com).

The IPv4 firewall can be configured to bypass certain attack checks if the initial packet in the flow matches a stateless IPv4 ACP entry. This bypass can be helpful in cases where it is known that certain traffic will trigger an attack condition in the firewall, but the condition is really a false positive and the traffic is known not to be a threat. The attack checks that can be bypassed using a stateless IPv4 ACP entry have been noted in *Table 1 on page 88*.

Refer to *Table 1 on page 88* for a complete list of the messages along with threat description, category, and weight. In addition, a notation is added next to the threats for which attack checking can be bypassed using a stateless IPv4 ACP entry.

Table A-1. Attack Log Messages

ID	Message and Description	Category and Weight
1	**TCP connection request received is invalid (expected SYN), dropping packet; flags=<*flags*>**	
	Short Definition: TCP: expected SYN **Description:** Indicates that the first packet in a TCP flow did not have the SYN flag set. The firewall maintains a state for each TCP flow and inspects the TCP flags to ensure that they are valid for the current state of the flow. The first packet of a TCP flow should have the SYN flag (and no other flags) set to indicate the beginning of the three-way handshake to transition from the LISTEN state to the SYN RCVD and SYN SENT states. This threat can be observed for valid traffic when the policy session is deleted or times out, but the TCP session is still established. Check your TCP policy timeout settings and verify that the timeout accounts for the longest interval between observing packets. **Action:** The firewall drops the offending packet.	**Category:** TCP Statefulness **Weight:** 6 **Check can be bypassed by using a stateless IPv4 ACP entry**
2	**TCP connection request received is invalid (expecting SYN only), dropping packet; flags=<*flags*>**	
	Short Definition: TCP: expected SYN only **Description:** Indicates that the first packet in a TCP flow had other flags set in addition to the SYN flag. The firewall maintains a state for each TCP flow and inspects the TCP flags to ensure that they are valid for the current state of the flow. The first packet of a TCP flow should have the SYN flag (and no other flags) set to indicate the beginning of the three-way handshake to transition from the LISTEN state to the SYN RCVD and SYN SENT states. This threat can be observed for valid traffic when the policy session is deleted or times out, but the TCP session is still established. Check your TCP policy timeout settings and verify that the timeout accounts for the longest interval between observing packets. **Action:** The firewall drops the offending packet.	**Category:** TCP Statefulness **Weight:** 7 **Check can be bypassed by using a stateless IPv4 ACP entry**
3	**TCP connection request received is invalid (expected SYN, got ACK), dropping packet; flags=<*flags*>**	
	Short Definition: TCP: expected SYN, got ACK **Description:** Indicates that the first packet in a TCP flow had the ACK flag set in addition to the SYN flag. The firewall maintains a state for each TCP flow and inspects the TCP flags to ensure that they are valid for the current state of the flow. The first packet of a TCP flow should have the SYN flag (and no other flags) set to indicate the beginning of the three-way handshake to transition from the LISTEN state to the SYN RCVD and SYN SENT states. This threat can be observed for valid traffic when the policy session is deleted or times out, but the TCP session is still established. Check your TCP policy timeout settings and verify that the timeout accounts for the longest interval between observing packets. **Action:** The firewall drops the offending packet.	**Category:** TCP Statefulness **Weight:** 6 **Check can be bypassed by using a stateless IPv4 ACP entry**

Table A-1. Attack Log Messages *(Continued)*

ID	Message and Description	Category and Weight
4	**TCP connection request received is invalid (expected SYN, got RST), dropping packet; flags=<*flags*>**	
	Short Definition: TCP: expected SYN, got RST **Description:** Indicates that the first packet in a TCP flow had the RST flag set in addition to the SYN flag. The firewall maintains a state for each TCP flow and inspects the TCP flags to ensure that they are valid for the current state of the flow. The first packet of a TCP flow should have the SYN flag (and no other flags) set to indicate the beginning of the three-way handshake to transition from the LISTEN state to the SYN RCVD and SYN SENT states. This threat can be observed for valid traffic when the policy session is deleted or times out, but the TCP session is still established. Check your TCP policy timeout settings and verify that the timeout accounts for the longest interval between observing packets. **Action:** The firewall drops the offending packet.	**Category:** TCP Statefulness **Weight:** 6 **Check can be bypassed by using a stateless IPv4 ACP entry**
5	**Received ACK from initiator but did not receive SYN/ACK from responder**	
	Short Definition: TCP: ACK before SYN/ACK **Description:** Indicates that a packet with only the ACK flag set was received from the initiator of the TCP flow even though a packet with both the SYN and ACK flags set has not yet been received from the responder. The firewall maintains a state for each TCP flow and inspects the TCP flags to ensure that they are valid for the current state of the flow. After a packet that only has the SYN flag set initiates the TCP flow, the next packet in the flow should be a SYN/ACK packet sent from the responder back to the initiator. It is only after the SYN/ACK packet has been received by the initiator that the initiator should send an ACK packet back to the responder. At this point the three-way handshake is complete and the TCP connection is fully established. Therefore, if an initiator sends out an ACK packet before it receives a SYN/ACK packet, this indicates either incorrect implementation of TCP in the initiator or that the initiator is an attacker. An attacker might, for example, be attempting a DoS attack, the intent being to establish many different TCP connections through the firewall by repeatedly sending SYN packets immediately followed by ACK packets (in the hope that the firewall does not simply drop the ACK packets) so that there is no room for other TCP connections to be established. If the network is configured for asymmetric routing, there is also a possibility that the initiator did receive a SYN/ACK packet before sending out the ACK packet, but that the firewall only saw the ACK packet because the SYN/ACK packet traveled a different path through the network. **Action:** The firewall drops the offending packet.	**Category:** TCP Statefulness **Weight:** 6 **Check can be bypassed by using a stateless IPv4 ACP entry**

Table A-1. Attack Log Messages *(Continued)*

ID	Message and Description	Category and Weight
6	**Post connection SYN attack detected**	
	Short Definition: Post connection SYN attack **Description:** Indicates that a packet with the SYN flag set was received for an established TCP connection. The firewall maintains a state for each TCP flow and inspects the TCP flags to ensure that they are valid for the current state of the flow. The SYN flag should not be received for an established TCP connection, indicating a possible attack. For example, an attacker could send a spoofed packet with the SYN flag set in order to have the legitimate client receive an RST packet, thus disrupting the connection. This threat can also be caused by an incorrect implementation of TCP in a client. For example, if a client does not change source ports between sessions and attempts to initiate a new session within the TIME WAIT timeout, this threat will be observed. **Action:** The firewall drops the offending packet.	**Category:** TCP Statefulness **Weight:** 7 **Check can be bypassed by using a stateless IPv4 ACP entry**
7	**Data packet received after reset, dropping packet**	
	Short Definition: RX data after TCP RST **Description:** Indicates the receipt of data on a TCP connection after a RST has already been received on that connection. The firewall maintains a state for each TCP flow and inspects the TCP flags to ensure that they are valid for the current state of the flow. This attack check prevents an attacker from sending data on a TCP connection that has already been closed. **Action:** The firewall drops the offending packet.	**Category:** TCP Statefulness **Weight:** 4 **Check can be bypassed by using a stateless IPv4 ACP entry**
9	**Invalid sequence number received with RST, dropping packet, seq=<*seq*>, high=<*high*>**	
	Short Definition: Invalid seq # with RST **Description:** Indicates the receipt of a TCP packet with the RST flag set whose sequence number is outside the valid range of sequence numbers. The firewall maintains a state for each TCP flow and inspects the sequence number to ensure that it is valid for the current state of the flow. **Action:** The firewall drops the offending packet.	**Category:** TCP Statefulness **Weight:** 6 **Check can be bypassed by using a stateless IPv4 ACP entry**
10	**Invalid ack value received for connection, dropping packet**	
	Short Definition: Invalid TCP ACK value **Description:** Indicates the receipt of a TCP packet with the ACK flag set whose acknowledgment number is invalid for the current state of the flow. The firewall maintains a state for each TCP flow and inspects the acknowledgment number to ensure that it is valid for the current state of the flow. **Action:** The firewall drops the offending packet.	**Category:** TCP Statefulness **Weight:** 7 **Check can be bypassed by using a stateless IPv4 ACP entry**

Table A-1. Attack Log Messages *(Continued)*

ID	Message and Description	Category and Weight
52	**ICMP error message received for uninitiated connection**	
	Short Definition: No session for ICMP error **Description:** Indicates the receipt of an ICMP error message for which no corresponding flow exists. The firewall determines the original flow corresponding to the ICMP error message, and if none exists, drops the offending packet. This threat could be observed for valid traffic if the policy session for the original flow times out or is deleted, or if the original flow has not been observed by the firewall due to asymmetric routing. **Action:** The firewall drops the offending packet.	**Category:** ICMP Statefulness **Weight:** 6 **Check can be bypassed by using a stateless IPv4 ACP entry**
100	**Zero length IP option detected**	
	Short Definition: Zero length IP option **Description:** Indicates the receipt of an IPv4 packet with an IP option length of zero. Some systems cannot handle packets of this type correctly. Because of this, an attacker could craft a packet of this type in order to crash vulnerable systems. **Action:** The firewall drops the offending packet.	**Category:** IP Options **Weight:** 7
101	**Source routing option set in IP packet**	
	Short Definition: Source routing option set **Description:** Indicates the receipt of an IPv4 packet with the source routing IP option set. The source routing IP option can be used maliciously in order to route packets through devices under the control of the attacker. **Action:** The firewall drops the offending packet.	**Category:** IP Options **Weight:** 4
150	**Zero bytes transferred for connection**	
	Short Definition: Connection with no data **Description:** Indicates that a policy session has been created and has timed out without any data being observed for that connection. This threat can be caused by a port scan. An attacker could send a TCP SYN message to many ports in order to determine whether there are any services listening on those ports with the intention of exploiting those services. Port scans can be prevented or limited in several ways: • Configure an IPv4 ACP to control port access. • Configure a limit to the number of sessions that can be opened by any one host. This can be configured on a per IPv4 ACP basis using the command **ip policy-class** <*ipv4 acp name*> **max-host-sessions** <*number*>. • Enable the stealth setting using the command **ip firewall stealth**. Refer to the *AOS Command Reference Guide* (available online at https://supportforums.adtran.com) for more information on this command.	**Category:** Timeout **Weight:** 2 **Check can be bypassed by using a stateless IPv4 ACP entry**

Table A-1. Attack Log Messages *(Continued)*

ID	Message and Description	Category and Weight
151	**Data connection not established from remote**	
	Short Definition: No connection from remote **Description:** Indicates that a pending policy session has timed out without being used. Pending policy sessions are typically created by ALGs to anticipate the reception of returning traffic. If a malicious user is purposely using an application to create openings through the firewall for malicious purposes, this could be an attack. In some cases, this is a valid message to receive. For example, the SIP ALG will create a pending policy session anticipating RTCP traffic. If the user agent never sends RTCP, then this policy session will never become active, resulting in one occurrence of this threat.	**Category:** Timeout **Weight:** 2
200	**Fragment of size less than configured minimum fragment size detected**	
	Short Definition: Fragment size < minimum **Description:** Indicates the receipt of a fragment smaller than the minimum allowed fragment size. This threat is detected during IP reassembly. It can indicate a corrupted or malformed packet, or it could indicate a possible attack. **Action:** The reassembly engine drops the offending fragment.	**Category:** Reassembly **Weight:** 7
201	**Tiny or overlapping fragment attack detected**	
	Short Definition: Tiny fragment attack **Description:** Indicates the receipt of an unusually small TCP fragment. An attacker could be attempting to bypass firewall rules by forcing a portion of the TCP header into subsequent fragments. Additionally, an attacker could be attempting to consume processing time on a server by sending it many very small packets to reassemble. **Action:** The reassembly engine drops the offending tiny fragment.	**Category:** Reassembly **Weight:** 10
202	**IpReasmbly datagram size exceeds max limit**	
	Short Definition: Datagram exceeds max size **Description:** Indicates the receipt of an IP datagram larger than the maximum allowable size of 65535 bytes. Some systems cannot handle datagrams of this type correctly. Because of this, an attacker could craft a datagram of this type in order to crash vulnerable systems. **Action:** The reassembly engine drops the offending datagram.	**Category:** Reassembly **Weight:** 7
203	**IpReasmbly last fragment length changed**	
	Short Definition: Last fragment length changed **Description:** Indicates an error in IP fragment reassembly. **Action:** The reassembly engine drops the offending fragment.	**Category:** Reassembly **Weight:** 6

Table A-1. Attack Log Messages *(Continued)*

ID	Message and Description	Category and Weight
204	**IpReasmbly Fragment count exceeds max limit**	
	Short Definition: Exceeded max fragments **Description:** Indicates the receipt of more than the maximum number of allowable fragments prior to reassembly. An attacker could have attempted to consume all of the resources in the reassembly engine of a server or other network device. **Action:** The reassembly engine drops the offending fragments.	**Category:** Reassembly **Weight:** 6
205	**IpReasmbly time out**	
	Short Definition: Reassembly timeout **Description:** Indicates that the fragments necessary to complete reassembly have not arrived in a timely manner. An attacker could have attempted to consume all of the resources in the reassembly engine of a server or other network device. **Action:** The reassembly engine drops the offending fragments and frees the resource.	**Category:** Reassembly **Weight:** 1
250	**Source IP is a broadcast address, dropping packet**	
	Short Definition: Source IP is broadcast **Description:** Indicates the receipt of an IPv4 packet whose source address is broadcast. An attacker could be attempting to initiate or propagate a DoS attack, possibly to an unknown host, by causing a vulnerable system to reply to the broadcast address. **Action:** The firewall drops the offending packet.	**Category:** Against Specifications **Weight:** 10
251	**TCP connection request received has invalid TCP header length, dropping packet**	
	Short Definition: Invalid TCP hdr length **Description:** Indicates the receipt of a TCP packet whose TCP header length is invalid. Some systems cannot handle packets of this type correctly. Because of this, an attacker could craft a packet of this type in order to crash or execute malicious code within vulnerable systems. **Action:** The firewall drops the offending packet.	**Category:** Against Specifications **Weight:** 7
252	**IP header length is less than the minimum length**	
	Short Definition: IP hdr length too small **Description:** Indicates the receipt of an IPv4 packet whose IP header length is smaller than the smallest allowable IP header. Some systems cannot handle packets of this type correctly. Because of this, an attacker could craft a packet of this type in order to crash or execute malicious code within vulnerable systems. **Action:** The firewall drops the offending packet.	**Category:** Against Specifications **Weight:** 7

Table A-1. Attack Log Messages *(Continued)*

ID	Message and Description	Category and Weight
253	**Packet without any data received**	
	Short Definition: Pkt w/o data received **Description:** Indicates the receipt of an IPv4 packet containing an IP header and no data. Some systems cannot handle packets of this type correctly. Because of this, an attacker could craft a packet of this type in order to crash vulnerable systems **Action:** The firewall drops the offending packet.	**Category:** Against Specifications **Weight:** 7
254	**Packet with Short TCP Header length detected, packet dropped**	
	Short Definition: Short TCP hdr length **Description:** Indicates the receipt of a TCP packet whose TCP header length is smaller than the smallest allowable TCP header. Some systems cannot handle packets of this type correctly. Because of this, an attacker could craft a packet of this type in order to crash or execute malicious code within vulnerable systems. **Action:** The firewall drops the offending packet.	**Category:** Against Specifications **Weight: 7**
255	**Dropping packet due to length problem in TCP**	
	Short Definition: Len in TCP hdr > pkt size **Description:** Indicates the receipt of a TCP packet whose TCP header length is larger than the actual packet. Some systems cannot handle packets of this type correctly. Because of this, an attacker could craft a packet of this type in order to crash or execute malicious code within vulnerable systems. **Action:** The firewall drops the offending packet.	**Category:** Against Specifications **Weight:** 7
256	**Packet with short UDP header length detected, packet dropped**	
	Short Definition: Short UDP hdr length **Description:** Indicates the receipt of a UDP packet whose UDP header length is smaller than the smallest allowable UDP header. Some systems cannot handle packets of this type correctly. Because of this, an attacker could craft a packet of this type in order to crash or execute malicious code within vulnerable systems. **Action:** The firewall drops the offending packet.	**Category:** Against Specifications **Weight:** 7
257	**Dropping packet because of invalid length in UDP**	
	Short Definition: Len in UDP hdr > pkt size **Description:** Indicates the receipt of a UDP packet whose UDP header length is larger than the actual packet. Some systems cannot handle packets of this type correctly. Because of this, an attacker could craft a packet of this type in order to crash or execute malicious code within vulnerable systems. **Action:** The firewall drops the offending packet.	**Category:** Against Specifications **Weight:** 7

Table A-1. Attack Log Messages *(Continued)*

ID	Message and Description	Category and Weight
258	**Packet with Short ICMP length detected, packet dropped**	
	Short Definition: Short ICMP hdr length **Description:** Indicates the receipt of an ICMP packet whose ICMP header length is smaller than the smallest allowable ICMP header. Some systems cannot handle packets of this type correctly. Because of this, an attacker could craft a packet of this type in order to crash or execute malicious code within vulnerable systems. **Action:** The firewall drops the offending packet.	**Category:** Against Specifications **Weight:** 7
259	**ICMP error message contains less data than expected (Possible attack)**	
	Short Definition: ICMP error data too small **Description:** Indicates the receipt of an ICMP error message with less data than expected. An ICMP error message is accompanied by information regarding the errored packet that caused it to be sent. This threat indicates that such information is missing or truncated. Some systems cannot handle ICMP error messages of this type correctly. Because of this, an attacker could craft an ICMP error message of this type in order to crash or execute malicious code within vulnerable systems. **Action:** The firewall drops the offending packet.	**Category:** Against Specifications **Weight:** 6
260	**HTTP Method exceeded max, dropping pkt src=<*ip address*>**	
	Short Definition: HTTP method > max size **Description:** Indicates the receipt of an HTTP packet whose method field exceeds the maximum allowable length. Some systems cannot handle packets of this type correctly. Because of this, an attacker could craft a packet of this type in order to crash or execute malicious code within vulnerable systems. This threat is only observed when URL filtering is being used. **Action:** The firewall drops the offending packet and sends TCP RST packets to both the client and the server to close the connection.	**Category:** Against Specifications **Weight:** 5
261	**HTTP URI exceeded max, dropping pkt src=<*ip address*>**	
	Short Definition: HTTP URI > max size **Description:** Indicates the receipt of an HTTP packet whose Uniform Resource Identifier (URI) field exceeds the maximum allowable length. Some systems cannot handle packets of this type correctly. Because of this, an attacker could craft a packet of this type in order to crash or execute malicious code within vulnerable systems. This threat is only observed when URL filtering is being used. **Action:** The firewall drops the offending packet and sends TCP RST packets to both the client and the server to close the connection.	**Category:** Against Specifications **Weight:** 5

Table A-1. Attack Log Messages *(Continued)*

ID	Message and Description	Category and Weight
262	**HTTP version exceeded max, dropping pkt src=<*ip address*>**	
	Short Definition: HTTP version > max size **Description:** Indicates the receipt of an HTTP packet whose version field exceeds the maximum allowable length. Some systems cannot handle packets of this type correctly. Because of this, an attacker could craft a packet of this type in order to crash or execute malicious code within vulnerable systems. This threat is only observed when URL filtering is being used. **Action:** The firewall drops the offending packet and sends TCP RST packets to both the client and the server to close the connection.	**Category:** Against Specifications **Weight:** 5
350	**Unable to determine route to destination, dropping packet**	
	Short Definition: No route found for dest **Description:** Indicates that no route was found for the destination of the first packet in a flow. The firewall performs a route lookup on all first packets in order to determine the destination ACP for the flow. This destination ACP is needed so that return traffic can also match the flow. Additionally, some ACP entries match against a destination ACP in order to enforce certain rules. This is commonly used in load sharing. Inspect your routing configuration to determine that correct routes exist for all valid traffic. **Action:** The firewall drops the offending packet.	**Category:** Route Table Issue **Weight:** 1
351	**Unable to find route for source, dropping packet**	
	Short Definition: No route found for source **Description:** Indicates that no route was found for the source of a packet. The firewall performs a route lookup on the source of a packet in order to ensure that the packet arrived on the correct ACP, which helps prevent traffic spoofing. Inspect your routing configuration to determine that correct routes exist for all valid traffic. **Action:** The firewall drops the offending packet.	**Category:** Route Table Issue **Weight:** 5
352	**Unable to guess source interface for ICMP error packet contents**	
	Short Definition: Bad src iface for ICMP **Description:** Indicates that upon a route lookup, no source interface was found for the errored packet provided with an ICMP error message. The firewall performs a route lookup on the errored packet corresponding to the received ICMP error message. This is used to ensure that the ICMP error message matches an existing flow, helping to prevent an attacker from sending a malicious uninitiated ICMP error message. This threat could indicate a problem with the routing configuration or an uninitiated ICMP error message. Inspect your routing configuration to determine that correct routes exist for all valid traffic. **Action:** The firewall drops the offending ICMP error message.	**Category:** Route Table Issue **Weight:** 4

Table A-1. Attack Log Messages *(Continued)*

ID	Message and Description	Category and Weight
353	**Unable to find source interface for incoming packet!**	
	Short Definition: Bad src iface for pkt **Description:** Indicates that upon a route lookup, no source interface was found for the packet. The firewall performs a route lookup on the source of a packet in order to ensure that the packet arrived on the correct ACP, which helps prevent traffic spoofing. Inspect your routing configuration to determine that correct routes exist for all valid traffic. **Action:** The firewall drops the offending packet.	**Category:** Route Table Issue **Weight:** 4
400	**Ceiling for per-host associations reached; dropping packet**	
	Short Definition: Max per-host sessions **Description:** Indicates that the maximum number of connections have been established from a given host. Each ACP can be configured to enforce a maximum number of connections that can be established from any given host. This can be configured using the command **ip policy-class** <*ipv4 acp name*> **max-host-sessions** <*number*>. (By default, no limits are enforced on the number of sessions established from a given host.) If this limit is reached, it can indicate a possible port scan, or it can indicate that an attacker is initiating or propagating a DoS attack with the intention of consuming all of the available network resources. You can view the current host sessions using the CLI by issuing the command **show ip policy-class** [<*ipv4 acp name*>] **host-sessions**. **Action:** Any traffic exceeding the limit is dropped until the number of connections drops below the configured limit.	**Category:** Configuration-related **Weight:** 9
401	**Security policy configured but plain pkt received, dropping packet**	
	Short Definition: Plain pkt on secure iface **Description:** Indicates that an unencrypted packet was received on a connection marked for encrypted traffic. If a connection is to be used for VPN, the connection is associated with the appropriate IPSec SA to enforce that only encrypted traffic will arrive from a flow associated with a VPN tunnel. This prevents unauthorized traffic from being injected into a secure network. Verify that the remote endpoint of the VPN tunnel is properly encrypting the traffic before sending it. **Action:** The firewall drops the unencrypted packet.	**Category:** Configuration-related **Weight:** 6
450	**General attack detected, dropping packet**	
	Short Definition: General attack detected **Description:** No detailed information is available regarding this threat. **Action:** The firewall drops the offending packet.	**Category:** Miscellaneous **Weight:** 8

Table A-1. Attack Log Messages *(Continued)*

ID	Message and Description	Category and Weight
451	**Packet with unsupported IP protocol received, dropping packet**	
	Short Definition: Unsupported IP protocol **Description:** Indicates the receipt of a packet with an unsupported IP protocol. In order to properly inspect the statefulness of traffic flows, the firewall drops any IP protocols that are not well known. If NAT is not required, an unsupported IP protocol can be allowed through the firewall statelessly using a stateless allow in the ACP. Supported IP protocols include: • AH • ESP • GRE • ICMP • IGMP • OSPF • PIM • TCP • UDP • VRRP **Action:** The firewall drops the offending packet.	**Category:** Miscellaneous **Weight:** 3 **Check can be bypassed by using a stateless IPv4 ACP entry**
452	**Dropping ICMP packet of type <*type*>**	
	Short Definition: Unsupported ICMP type **Description:** Indicates the receipt of an ICMP packet with an unsupported type. In order to properly inspect the statefulness of ICMP traffic flows, the firewall drops any ICMP types that are not well known. If NAT is not required, an unsupported ICMP type can be allowed through the firewall statelessly using a stateless allow in the ACP. Supported ICMP types include the following: • Echo • Timestamp • Destination unreachable • Source quench • Time-to-live (TTL) exceeded **Action:** The firewall drops the offending packet.	**Category:** Miscellaneous **Weight:** 4 **Check can be bypassed by using a stateless IPv4 ACP entry**

Table A-1. Attack Log Messages *(Continued)*

ID	Message and Description	Category and Weight
550	**Spoofing detected, dropping packet**	
	Short Definition: Spoofing detected **Description:** Indicates the receipt of a packet on a different ACP than determined by a route lookup on the source of the packet. The firewall performs a route lookup on the source of packets to determine whether they have arrived on the correct ACP. A packet arriving on a different ACP than indicated by a route lookup may be spoofed. In certain routing configurations (for example, when policy-based routing (PBR) can act on certain packets of the flow), you might want to allow traffic to arrive on an ACP that differs from the results of the route lookup. If this is desired, use the command **no ip policy-class** <*ipv4 acp name*> **rpf-check** to disable the reverse path forwarding check for packets arriving at that ACP. **Action:** The firewall drops the offending packet.	**Category:** Spoofing **Weight:** 8
551	**Blind Spoofing attack detected**	
	Short Definition: Blind spoofing attack **Description:** Indicates the receipt of a TCP packet with the FIN flag set without the ACK flag. **Action:** The firewall drops the offending packet.	**Category:** Spoofing **Weight:** 8
552	**Echo response for uninitiated echo request (possible smurf attack), dropping packet**	
	Short Definition: Possible ICMP smurf attack **Description:** Indicates the receipt of an ICMP echo response for which no ICMP echo request was observed by the firewall. The firewall maintains the state of an ICMP connection and will attempt to match a response to its original request. If no corresponding request is found, the response is dropped. An uninitiated response can indicate a possible smurf attack, in which an attacker spoofs an ICMP echo request, typically to a broadcast address. The source of the request is spoofed to be the address of a legitimate server or other network resource, causing all of the hosts in the broadcast domain to flood the legitimate resource with responses. **Action:** The firewall drops the offending packet.	**Category:** Smurf Attack **Weight:** 9 **Check can be bypassed by using a stateless IPv4 ACP entry**
553	**UDP echo response received for uninitiated echo request (possible smurf attack), dropping packet**	
	Short Definition: Possible UDP smurf attack **Description:** Indicates the receipt of a UDP echo response for which no UDP echo request was observed by the firewall. The firewall maintains the state of a UDP connection and will attempt to match a response to its original request. If no corresponding request is found, the response is dropped. An uninitiated response can indicate a possible smurf attack, in which an attacker spoofs a UDP echo request, typically to a broadcast address. The source of the request is spoofed to be the address of a legitimate server or other network resource, causing all of the hosts in the broadcast domain to flood the legitimate resource with responses. **Action:** The firewall drops the offending packet.	**Category:** Smurf Attack **Weight:** 9 **Check can be bypassed by using a stateless IPv4 ACP entry**

Table A-1. Attack Log Messages *(Continued)*

ID	Message and Description	Category and Weight
554	**TCP Null Scan attack detected**	
	Short Definition: TCP Null scan attack **Description:** Indicates the receipt of a TCP packet with no flags set and a sequence number of zero. Since this packet provides no state information or data, its sole purpose is the anticipation of a response. This type of packet is used in a port scan to determine what TCP ports are open and whether services are listening on those ports. An attacker could use this information to later exploit a resource at that port. **Action:** The firewall drops the offending packet.	**Category:** TCP Null Scan Attack **Weight:** 8
555	**Crossed 80% of associations allocated. Possible TCP SYN flooding.**	
	Short Definition: Possible TCP SYN flood **Description:** Indicates that 80 percent of the maximum configured connections for an ACP are in use. **Action:** The firewall performs SYN flooding prevention using SYN cookies until the number of connections drops below 70 percent.	**Category:** TCP Syn Flood Attack **Weight:** 8
556	**Possible Land attack detected, dropping packet**	
	Short Definition: Possible Land attack **Description:** Indicates the receipt of a packet whose source and destination information is identical. An attacker could craft a packet of this type in order to exploit vulnerabilities in systems that do not handle these packets correctly. The packet is crafted so that the source and destination of the packet are an open port on a network resource. This will cause the vulnerable system to reply to itself continuously. **Action:** The firewall drops the offending packet.	**Category:** Land Attack **Weight:** 10
557	**Length in IP Header > Data length. Possible JOLT attack**	
	Short Definition: Possible JOLT attack **Description:** Indicates the receipt of fragments for which the total length reported in the IPv4 header is longer than the actual length of the reassembled packet, indicating a possible JOLT attack. Some systems cannot handle fragments of this type correctly. Because of this, an attacker could craft a fragment of this type in order to crash vulnerable systems or make them unresponsive. **Action:** The reassembly engine drops the offending fragments.	**Category:** JOLT Attack **Weight:** 10

Table A-1. Attack Log Messages *(Continued)*

ID	Message and Description	Category and Weight
558	**Ping of Death attack detected**	
	Short Definition: Ping of Death attack **Description:** Indicates the receipt of fragments whose total reassembled length exceeds 65535 bytes, the maximum length for an IPv4 packet. Because older operating systems often permitted a user to send ICMP echo requests with this characteristic, this attack is known as the ping of death. Some systems cannot handle fragments of this type correctly. Because of this, an attacker could craft fragments of this type in order to crash vulnerable systems or make them unresponsive. **Action:** The reassembly engine drops the offending fragments.	**Category:** Ping of Death **Weight:** 10
559	**Possible TARGA3 attack: UDP length > IP length - IP header length.**	
	Short Definition: Possible TARGA3 attack **Description:** Indicates the receipt of a packet or fragment matching the characteristics of a TARGA3 attack. The TARGA3 attack encompasses a wide variety of protocols, header options, offsets, invalid fragmentation, etc., expecting that systems could be vulnerable to at least some of the packets in the suite. **Action:** The reassembly engine drops the offending fragments.	**Category:** TARGA3 Attack **Weight:** 10
560	**Terminating connection as WinNuke Attack detected, OOB packet**	
	Short Definition: WinNuke attack **Description:** Indicates the receipt of a TCP packet with the urgent (URG) flag set. Sometimes such traffic is referred to as OOB traffic. Certain systems could be vulnerable to this attack because they cannot process the URG flag correctly. An attacker could craft a packet with this flag set in order to exploit this vulnerability. **Action:** The firewall drops the offending packet.	**Category:** WinNuke Attack **Weight:** 9
562	**FTP PORT misdirection attack**	
	Short Definition: FTP PORT misdirection **Description:** Indicates the receipt of an FTP PORT command that specifies a port below 1024. This threat is similar to an FTP bounce attack. An attacker can use the FTP PORT command to obtain access to low numbered ports on the same network resource as the FTP server. These will often be ports to which the attacker would not ordinarily have access, and the attacker could possibly access other services listening at those ports. Note that if the FTP ALG is disabled, this threat will not be detected. **Action:** The firewall drops the offending packet.	**Category:** FTP Port Misdirection **Weight:** 8