August 2008

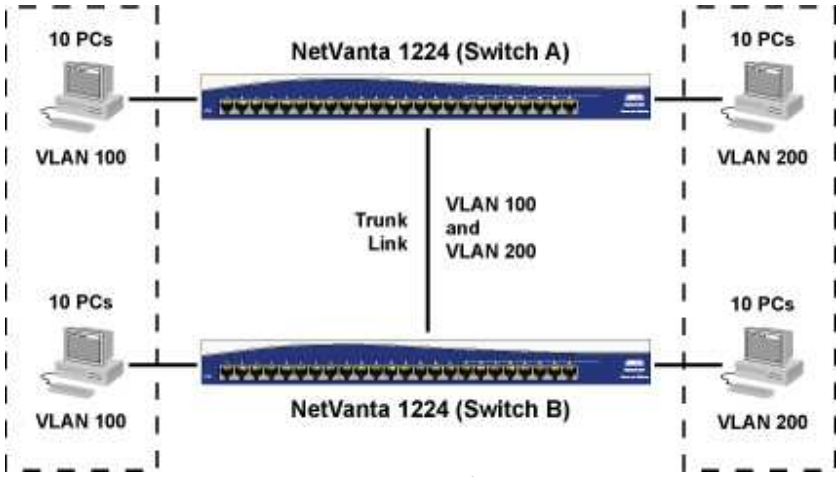
AOS Quick Configuration Guide
Configuring InterVLAN Routing in AOS

Understanding VLANs

A virtual local area network (VLAN) allows creation of logical subnetworks, broadening the ability to segment the networks in ways independent of the physical setup. VLANs have all the same attributes as traditional physical LANs, but allow network devices to be grouped together based on organizational function and application rather than be constrained by geographical or physical location. By creating VLANs, a switched network can consist of multiple segments, each with its own separate broadcast and multicast domains. You can set up VLANs either statically (where each switch interface is assigned specifically to a VLAN) or dynamically (based on MAC addresses).

Incorporating VLANs into a typical network provides benefits including security, broadcast or congestion control, and management. Through the use of VLANs, users can be isolated from one another; that is, a user in one VLAN cannot access data in a different VLAN. Also, just as switches isolate collision domains, VLANs isolate broadcast (messages sent to all users) and multicast (messages sent to a group of users) domains. By preventing broadcast and multicast traffic from traversing the entire network, network performance improves. The logical grouping of users also allows for easier network management. A network administrator can easily move an individual from one group to another without having to recable the network.

VLANs can span multiple switches. For example, you could have Ports 1 through 10 of Switch A assigned to VLAN 100, and Ports 11 through 20 of Switch A assigned to VLAN 200. If Switch A and Switch B share a high-speed link (i.e., are connected via a Gigabit Ethernet trunk link), then Switch B could also have ports assigned to the same VLANs as Switch A. See Figure 1 for a graphical depiction of this concept.

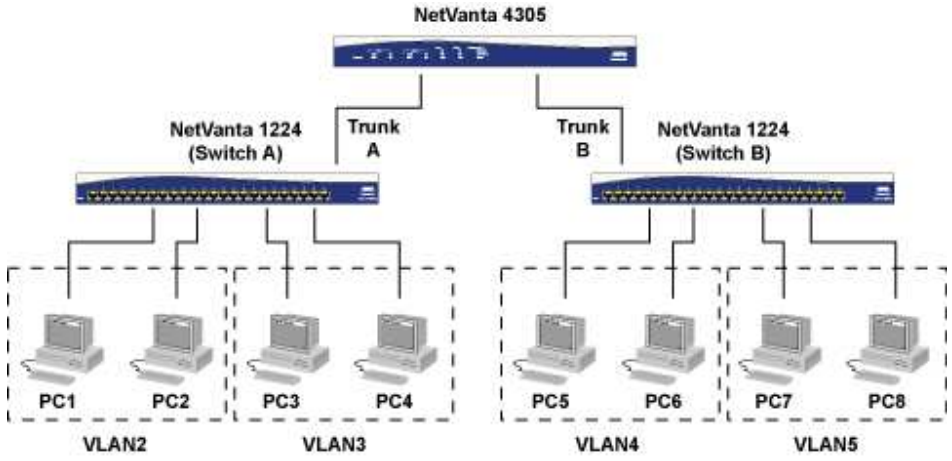


Basic VLAN Configuration
Figure 1

Understanding InterVLAN Routing

In order for network devices in different VLANs to communicate, a router must be used to route traffic between the VLANs. While VLANs help to control local traffic, if a device in one VLAN needs to communicate with a device in another VLAN one or more routers must be used for InterVLAN communication.

Figure 2 shows a topology where InterVLAN routing would be necessary for PCs in one VLAN to communicate with PCs in other VLANs. The router has two interfaces with 802.1Q encapsulation enabled and multiple VLANs configured on each. For PC1 in VLAN2 to communicate with PC2 in the same VLAN, PC1 simply sends a packet addressed to PC2. The switch will forward the packet directly to the destination PC without going through the router. However, for PC1 to send a packet to PC5, the switch will have to place a VLAN2 tag on the packet and forward the packet on Trunk A to the router. The router will remove the VLAN2 tag, determine the appropriate outgoing interface based on the IP route table, place a VLAN4 tag on the packet, and send it out on Trunk B. The switch in VLAN4 that receives the packet will forward it directly to PC5.



InterVLAN Routing Topology
Figure 2

Hardware/Software Requirements/Limitations

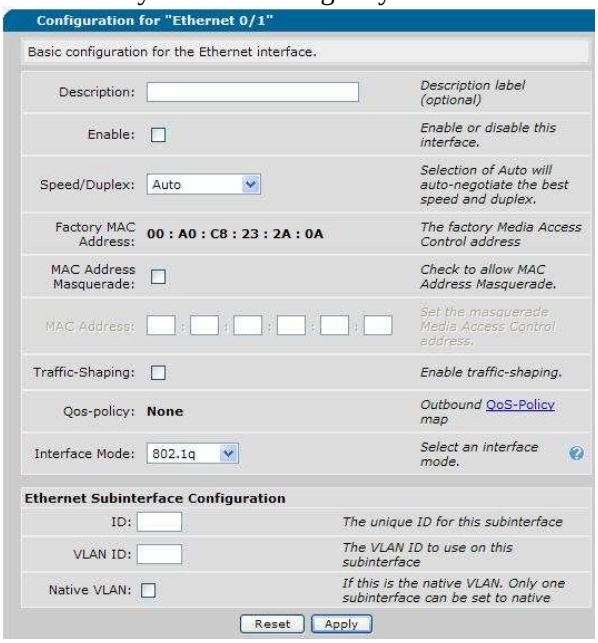
- Any AOS router is capable of supporting an 802.1Q trunk port.
- To configure standalone InterVLAN routing, a integrated switch router product such as a NetVanta 1224R, 1335, 6355, 3120, 3130, 3448, or 7100 is required.

Configuring 802.1Q Trunking

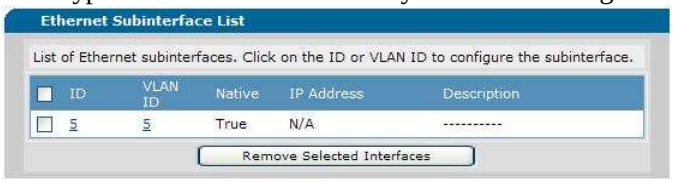
Configuring 802.1Q Trunking in the GUI

First, navigate to “Physical Interfaces”, and click the interface that you would like to convert to an 802.1Q trunk.

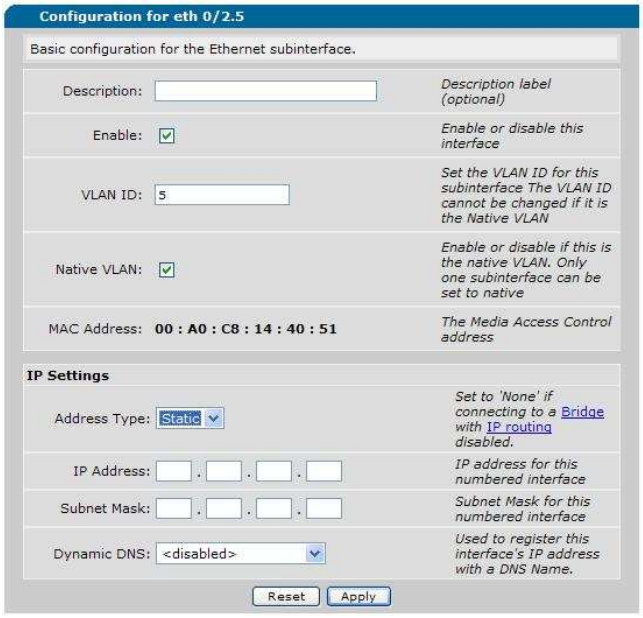
Next, select “802.1q” under the interface mode. Upon selecting this, the screen will change to allow you to enter VLAN IDs. Enter the VLANs that you will be using on your switches.



Once an interface is created, it will appear in the list of subinterfaces. To configure a subinterface, click on the hyperlinked “VLAN ID” that you wish to configure.



Finally, the subinterface configuration page allows you to edit the description, VLAN ID, and IP address of the VLAN. You can also select the Native VLAN.



Configuring 802.1Q Trunking in the CLI

1. First, access the interface that will be serving as the 802.1Q trunk. Then, enable 802.1q mode on the interface. Enable the interface by issuing “no shutdown”.

   Syntax: **encapsulation 802.1q**
   Syntax: **no shutdown**

2. Once 802.1Q mode has been enabled on an Ethernet interface, you will be allowed to create multiple subinterfaces. Each subinterface will be the default gateway for devices within that VLAN.

   Syntax: **interface** <*interface*> **0/1.1**

3. Each subinterface must be assigned a VLAN ID. This ID represents the VLAN number that the subinterface is responsible for. Valid IDs fall in the range of 1 to 4094. At least one subinterface must be specified as the “native” VLAN. This VLAN is where all untagged traffic will be tagged to.

   Syntax: **vlan-id** <*vlan number*>
   Syntax: **vlan-id** <*vlan number*> **native**

4. Finally, each subinterface must be assigned an IP address. This IP must reside in the desired subnet for the VLAN that has been assigned to it. After the assignment, the interface can be activated by issuing “no shutdown”.

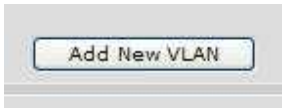
   (config-eth 0/1.1)# **ip address** <*A.B.C.D*> <*subnet mask*>
   (config-eth 0/1.1)# **no shutdown**

Configuring Standalone InterVLAN Routing

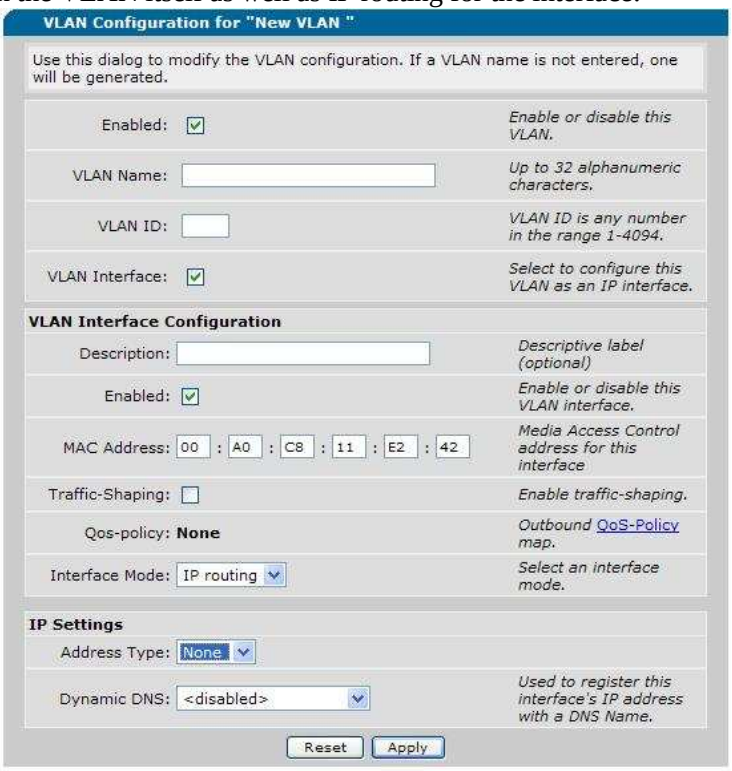
Since the integrated Router/Switch products do InterVLAN routing automatically, the only tasks that must be accomplished to allow communication is the creation of a VLAN interface for each VLAN and IP address assignment.

Configuring Standalone InterVLAN in the GUI

First, navigate to the "VLANs" menu and click the button to add a new VLAN.



Next, fill in the appropriate information for the VLAN you wish to create. Check both "Enable" boxes on this page to enable both the VLAN itself as well as IP routing for the interface.



Configuring Standalone InterVLAN in the CLI

1. A VLAN interface can be created by simply attempting to access the interface from the CLI. Once created, activate the interface by issuing "no shutdown".

   Syntax: **interface vlan** *<vlan number>*
   *Syntax***: no shutdown**

2. Next, assign an IP address to the VLAN interface. The IP address assigned to this interface will be the default gateway for the devices in the subnet.

   Syntax: **ip address** *<A.B.C.D> <subnet mask>*

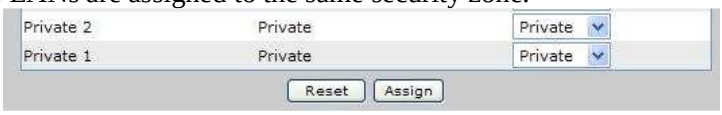
Configuring InterVLAN Firewall Rules

In a situation where a firewall is in place, either simply for security or to NAT traffic, additional rules will have to be created in order to allow VLANs to communicate. This section will provide a brief summary of how to create those rules.

The following configurations involve two VLANS, in subnets 10.11.1.0/24 and 10.11.2.0/24.

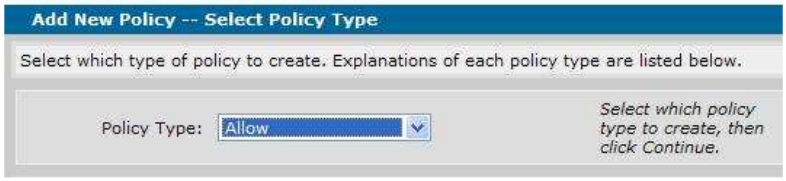
Configuring InterVLAN Firewall Rules in the GUI
First, navigate to “Security Zones”

Next, verify that both VLANs are assigned to the same security zone.

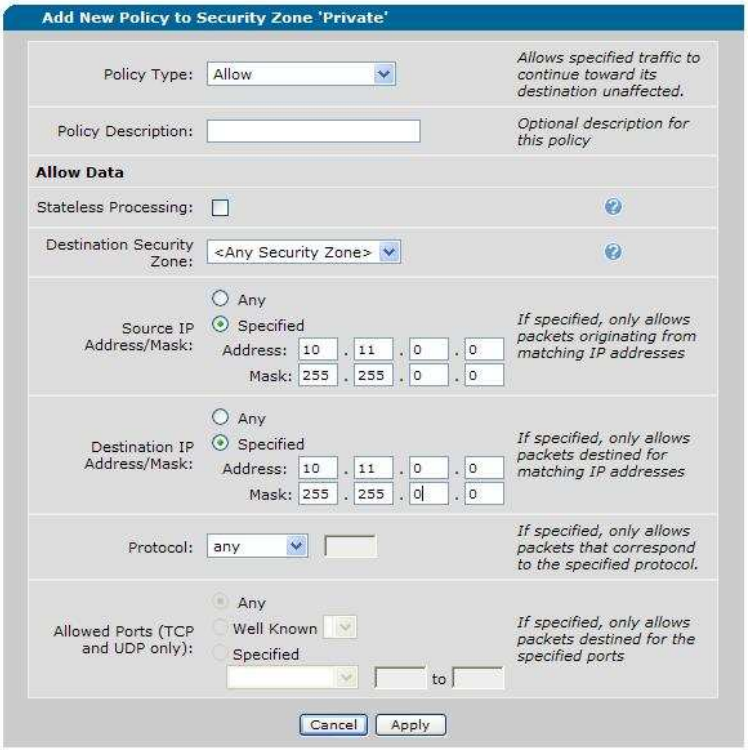


Next, click the link under “Edit Security Zones” for the security zone encompassing the VLANs. Once there, click the “Add Policy” button.

Select “Allow” for policy type and click the “Continue” button at the bottom of the page.

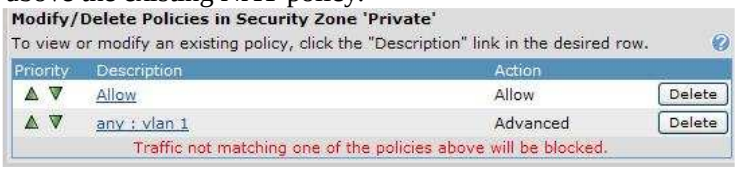


The settings for the allow policy will differ based on how your VLANs are subnetted. For example, VLANs such as 10.11.1.0/24 and 10.11.2.0/24 can be “supernetted”. This means that you can expand the subnet to encompass both of them. In this example, the subnet has been expanded to 10.11.0.0/16. This will cover traffic from either subnet destined for the other.



If your VLANs differ significantly (e.g. 192.168.1.0/24 and 172.16.1.1/24, then you will need to create 2 different allow policies. One policy will have a source of 192.168.1.0/24 and a destination of 172.16.1.0/24, while the second policy will have the reverse.

Finally if an existing policy is in place to NAT traffic to the internet, use the green arrows to ensure that the new "Allow" policy is above the existing NAT policy.



Configuring InterVLAN Firewall Rules in the CLI

1. Ensure that both VLANs are in the same security policy. This can be verified by issuing "show run" from the enable prompt. If each VLAN interface shows the same access policy, proceed to step 2. Otherwise, assign them to the same policy.

   Syntax: **access-policy** *<policy name>*
   Syntax: **access-policy** *<policy name>*

2. Create an access list to define traffic. The settings for the access list will differ based on how your VLANs are subnetted. For example, VLANs such as 10.11.1.0/24 and 10.11.2.0/24 can be "supernetted". This means that you can expand the subnet to encompass both of them. In this example, the subnet has been expanded to 10.11.0.0/16. This will cover traffic from either subnet destined for the other.

a. If your VLANs differ significantly (e.g. 192.168.1.0/24 and 172.16.1.1/24, then you will need to create 2 different allow policies. One policy will have a source of 192.168.1.0/24 and a destination of 172.16.1.0/24, while the second policy will have the reverse.

Syntax: **ip access-list extended** <*ACL name*>
Syntax: **permit ip** <*source net*> <*wildcard mask*> <*destination net*> <*wildcard mask*>

3. Finally, navigate to the policy that encompasses both VLANs. The allow list will need to be above any existing NAT statements in order to function. In the CLI, you must remove existing statements and add them back in the order you wish them to be executed. This can be accomplished by issuing "show run" to view the current statements and then issuing "no" along with whatever current statement is there. Next, add the statements back in, starting with the allow policy.

   Syntax: **ip policy-class** <*policy name*>
   Syntax: **allow list** <*ACL name*>

Command Summary Table

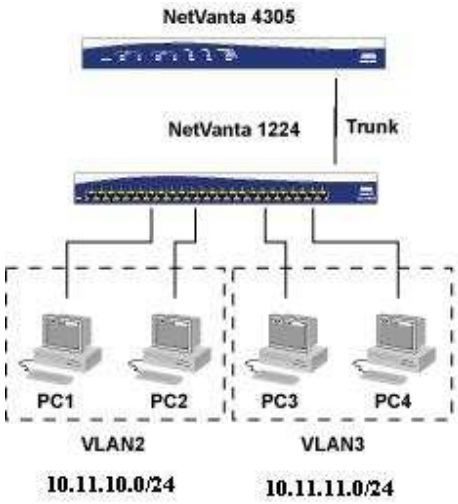
802.1Q Trunking

	Command	Description
Step 1	(config)# **interface eth** <*number*>	Access the interface in which trunking will be enabled
Step 2	(config eth 0/1)# **encapsulation 802.1q**	Enable 802.1Q encapsulation
Step 3	(config-eth 0/1)# **int eth** <*number*>.<*subinterface number*>	Create an Ethernet subinterface
Step 4	(config-eth 0/1.1)# **vlan-id** <*vlan number*>	Assign a VLAN to the subinterface
Step 5	(config-eth 0/1.1)# **ip address** <*ip address*> <*subnet mask*>	Assign an IP to the interface within the subnet of the VLAN
Step 6 (Optional)	(config-eth 0/1.1)# **vlan-id** <*vlan number*> **native**	Specify the VLAN subinterface as the native VLAN
Step 7	(config-eth 0/1.1)# **no shutdown**	Enable the interface

Standalone InterVLAN Routing

	Command	Description
Step 1	(config)# **interface vlan** <*number*>	Create a VLAN interface
Step 2	(config-vlan x)# **ip address** <*ip address*> <*subnet mask*>	Assign an IP address to the VLAN interface
Step 3	(config-vlan *x*)# **no shutdown**	Enable the VLAN interface

Example configuration

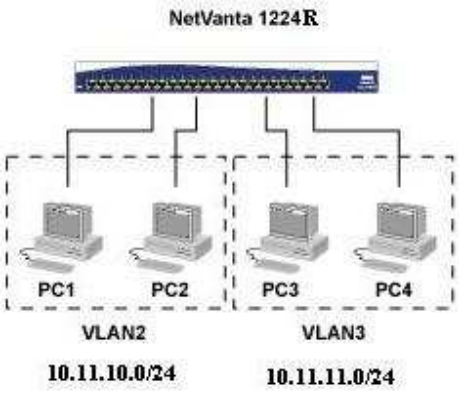


Ex. 1: 802.1Q Trunking

(config)#int eth 0/1
(config-eth 0/1)# encapsulation 802.1q
(config-eth 0/1)# int eth 0/1.2
(config-eth 0/1.2)# vlan-id 2
(config-eth 0/1.2)# vlan-id 2 native
(config-eth 0/1.2)# ip address 10.11.10.1 255.255.255.0
(config-eth 0/1.2)# no shutdown
(config-eth 0/1.2)# int eth 0/1.3
(config-eth 0/1.3)# vlan-id 3
(config-eth 0/1.3)# ip address 10.11.11.1 255.255.255.0
(config-eth 0/1.3)# no shutdown

Ex 2: 802.1Q Trunking w/Firewall Rules

(config)#int eth 0/1
(config-eth 0/1)# encapsulation 802.1q
(config-eth 0/1)# int eth 0/1.2
(config-eth 0/1.2)# vlan-id 2
(config-eth 0/1.2)# vlan-id 2 native
(config-eth 0/1.2)# ip address 10.11.10.1 255.255.255.0
(config-eth 0/1.2)# access-policy Private
(config-eth 0/1.2)# no shutdown
(config-eth 0/1.2)# int eth 0/1.3
(config-eth 0/1.3)# vlan-id 3
(config-eth 0/1.3)# ip address 10.11.11.1 255.255.255.0
(config-eth 0/1.3)# access-policy Private
(config-eth 0/1.3)# no shutdown
(config-eth 0/1.3)# ip access-list extended VLAN
(config-ext-acl)# permit ip 10.11.0.0 0.0.255.255 10.11.0.0 0.0.255.255
(config-ext-acl)# ip policy-class Private
(config-policy-class)# allow list VLAN



Ex 3: Standalone VLAN Routing

(config)#int vlan 2
(config-vlan 2)# ip address 10.11.10.1 255.255.255.0
(config-vlan 2)# no shutdown
(config-vlan 2)# int vlan 3
(config-vlan 2)# ip address 10.11.11.1 255.255.255.0
(config-vlan 2)# no shutdown

Ex 4: Standalone VLAN Routing w/Firewall Rules

(config)#int vlan 2
(config-vlan 2)# ip address 10.11.10.1 255.255.255.0
(config-vlan 2)# access-policy Private
(config-vlan 2)# no shutdown
(config-vlan 2)# int vlan 3
(config-vlan 3)# ip address 10.11.11.1 255.255.255.0
(config-vlan 3)# access-policy Private
(config-vlan 3)# no shutdown
(config-vlan 3)# ip access-list extended VLAN
(config-ext-acl)# permit ip 10.11.0.0 0.0.255.255 10.11.0.0 0.0.255.255
(config-ext-acl)# ip policy-class Private
(config-policy-class)# allow list VLAN

Troubleshooting

To see if the interface is up, use the command:

Syntax: **show interface** <interface type> <interface number>

```
vlan 1 is UP
  Hardware address is 00:A0:C8:23:2A:09
  ARP type: ARPA; ARP timeout is 20 minutes
  5 minute input rate 240 bits/sec, 0 packets/sec
  5 minute output rate 112 bits/sec, 0 packets/sec
    8455 packets input, 2035629 bytes
    6722 unicasts, 1733 broadcasts, 0 multicasts input
    6721 packets output, 988006 bytes
    4504 unicasts, 2217 broadcasts, 0 multicasts output
    0 discards
```

If both interfaces are up, verify connectivity by using a source ping. A source ping allows you to specify the source of the ping packet as well as the destination. Use one VLAN as the source and another as the destination.

Syntax: **ping** <destination IP> **source** <source IP>

To test the firewall, use the following command to see if sessions exist between local subnets:

Syntax: **show ip policy-sessions** <policy name>

```
Protocol (TTL) [in crypto map] -> [out crypto map] Destination policy-class
  Src IP Address  Src Port Dest IP Address Dst Port NAT IP Address    NAT Port
    ---------------      --------      ---------------       --------      -----------------       --------

Policy class "Private":
```