Technical Guide

Policy-based Routing

Feature Overview and Configuration Guide

Introduction

Policy-based routing (PBR) provides a means to route particular packets to their destination via a specific next-hop.

Using policy-based routing, you can control which packets follow which path through the network. The specific path that these packets will take can be based on configurable parameters such as priority, address, protocol, or VLAN membership.

Many AlliedWare Plus switches support policy-based routing at wire speed, using hardware-based policy routing facilities. AlliedWare Plus firewalls and routers support policy-based routing in software.

These products provide flexible options for configuring policy-based routing, and can support multiple different policy routes, with different next-hops, simultaneously. Possible benefits include both QoS and cost savings:

- QoS by using dedicated links for certain types of traffic.
- cost savings by splitting traffic between low-bandwidth, low-cost permanent paths and high-bandwidth, high-cost use-on-demand paths.

A data network is a significant investment. It is a tool that can add significant value to a business. In order to reap the maximum benefit from this investment, you need to have as much control as possible over the traffic that is flowing in the network.

Often straight-out destination-based IP routing does not provide the level of traffic-direction control that a network manager really needs. Policy-based routing provides a much finer-grained level of control over where packets are routed to.

Products and software version that apply to this guide

This guide applies to AlliedWare Plus™ switches, firewalls and routers that support policy-based routing, running version **5.4.5-2** or later. To see whether your device supports policy-based routing, see the following documents:

- The product's Datasheet
- The product's Command Reference

These documents are available from the above links on our website at alliedtelesis.com.

Related documents

The following documents give more information about the IP routing features on AlliedWare Plus products:

- Route Selection Feature Overview and Configuration Guide
- IP Addressing and Protocols Feature Overview and Configuration Guide
- the Command Reference for each product

These documents are available from the links above or on our website at alliedtelesis.com

Contents

Configuring Policy-based Routing on the Switch Products

On the switch products, policy-based routing is presented as an action in a QoS policy. It can be combined with other QoS actions performed in the same policy. Other QoS actions applied to the traffic will be carried out before the traffic is sent to its next-hop.

The selection of packets to be policy routed is carried out by the standard class-map packet matching rules. If the switch doesn't have the configured next-hop in its ARP table, it will send an ARP request for it. If it does not receive a reply, the switch will send an ICMP destination unreachable message to the originating host. The switch does not use the configured default route if the policy-based routing next-hop is unavailable.

So, the steps in configuring PBR on the switches are:

Step 1: For the SwitchBlade x8100 only, arrange the hardware tables to allocate some space for policy-based routes

To achieve this, enter the command:

```
awplus(config)# platform pbr-enable
```

Then the switch must be rebooted for the command to take effect.

Note: No other switch products need this command to be entered. All the other switches support PBR by default.

Step 2: Enable QoS

```
awplus(config)# mls qos enable
```

Step 3: Create a class-map that defines the match criteria for the traffic that is to be directed down the policy route

```
awplus(config)# class-map <cmap-name>
awplus(config-cmap)# match <match-criteria>
```

Step 4: Create a QoS policy, and define a policy routing rule for the traffic that matches the class-map

A policy routing rule simply consists of defining the next-hop via which the traffic will be routed.

```
awplus(config)# policy-map <pmap-name>
awplus(config-pmap)# class <cmap-name>
awplus(config-pmap-c)# set ip next-hop <nexthop-address>
```

Step 5: Apply the QoS policy to a port

```
awplus(config)# interface port1.0.1
awplus(config-if)# service-policy input <pmap-name>
```

Limitations of PBR on switches

Limited to IPv4 only

On the switch products, policy-based routing is only available for IPv4 traffic. Policy based routing of IPv6 is not available on any of the switch products.

Limit on the number of rules

An all switch products, the total number of policy-based rules is limited to 128.

Limitations on the main IPv4 hardware routing table when PBR is enabled on SBx8100

When the switch is booted up with the command **platform pbr-enable** in the startup script, then the silicon resources are allocated in such a way that the size of the hardware IPv4 unicast routing table is reduced by 512 entries.

This is not a severe limitation, as the hardware IPv4 routing table contains thousands of entries. The lowest number of entries is 5376, when the switch is configured with the default silicon profile and the platform routing ratio with IPv4 and IPv6 weighting balanced.

Configuring Policy-based Routing on AlliedWare Plus firewalls and routers

On an AlliedWare Plus firewall or router:

- Policy Based Routing is supported for both IPv4 and IPv6
- Multiple next-hops can be defined on each policy rule, with the first available next-hop being the one that is used
- If no next-hops are available, the traffic is not dropped, but instead is forwarded via the normal routing table

First, policy-based routing needs to be globally enabled;

```
AR4050S#configure terminal
AR4050S(config)#policy-based-routing
AR4050S(config-pbr)#policy-based-routing enable
AR4050S(config-pbr)#
```

Then a set of policy-routing rules are created.

The rules are of the form:

```
ip policy-route <ID> match <application> from <entity> to <entity> nexthop
<list-of-next-hops>
```

or

```
ipv6 policy-route <ID> match <application> from <entity> to <entity>
nexthop <list-of-next-hops>
```

Where:

The <*ID*> is a identifier for the rule. Packets are matched against the rules in order of ascending ID. Whether you specify the ID number for a rule is optional. If an ID number is not specified, then they are automatically allocated in intervals of 10.

The <*application*> is optional. It can be any predefined application, or application created by using the application command to enter application configuration mode. If the application parameter is not specified, then the rule matches any traffic type.

The to and from <*entities*> are optional. They can be any entities that have been defined by the zone, network or host commands. If the 'from' entity is not specified, then the rule matches traffic from any source. Similarly, if the 'to' entity is not specified, the rule matches traffic to any destination.

From version 5.5.2-1.1 onwards, you can use the tab key to auto-complete application and entity names. This makes it easier to specify the name of an existing DPI application or firewall entity.

The list of next-hops can be up to eight next-hops. The next-hops can be any one of:

- IPv4 addresses (except on dynamic interfaces such as PPPoE interfaces)
- IPv6 addresses (except on dynamic interfaces such as PPPoE interfaces)
- Interfaces

All the entries in the list must be of the same type—they can't be a mixture of the three types. Of course, if the policy route is matching IPv4 traffic (i.e. is a rule configured with **ip policy-route…**), then the next-hops must be IPv4 addresses or interfaces. If the policy route is matching IPv6 traffic (i.e. is a rule configured with **ipv6 policy-route…**) then the next-hops must be IPv6 addresses or interfaces.

The option of using interfaces is for the case where the policy route is directed over a tunnel or a PPP link, where the next-hop is not relevant, or maybe not even known. In that situation you cannot use an IP or IPv6 address; you must use the interfaces instead.

The next-hops in the list are chosen in turn until one is found to be available. If none of the next-hops in the list is available, then the traffic is just conventionally routed by entries in the device's main IP route table.

Some example rules

```
AR4050S#configure terminal
AR4050S(config)#policy-based-routing
AR4050S(config-pbr)#ip policy-route match udp from inside to lan nexthop 10.1.1.2
10.1.2.2 10.1.3.2 10.1.4.2 10.1.5.2 10.1.6.2 10.1.7.2 10.1.8.2
AR4050S(config-pbr)#ipv6 policy-route match udp from inside to lan nexthop
2001:100::2
```

Limitations

Limit on the total number of rules

The limit on the total number of Policy rules is 128. This number is spread across IPv4 and IPv6 rules. So, the total number of IPv4 rules, plus the total number of IPv6 rules, cannot exceed 128.

Limit on the number of combinations of next-hops

There is a limitation on the number of next-hops that can be shared among the rules. The limit is a total of **14** different combinations of next-hops, spread over the IPv4 rules and the IPv6 rules.

The same set of next-hops, but arranged in a different order, is considered a different combination.

So, for example the following lists of next-hops would count as two out of the possible 14 combinations:

```
128.34.56.182, 176.73.231.93

176.73.231.93, 128.34.56.182
```

Variation of DSCP within a flow will not be taken account of

One of the parameters that can be used to define an application is the DSCP value in the packets. The definition of an application can include a specification that packets belonging to the application must have a particular DSCP value, or one of a list of DSCP values. Once the first packet of a stream has been matched against a rule, then any subsequent packets in the same stream are not matched against the rules again (for performance reasons).

If the first packet in a stream has a particular DSCP value, and so matches a given application, and thereby, a particular rule, then ALL packets in the same stream will be deemed to match that same rule, even if they have different DSCP values that don't match the definition of the application.

Show Commands on AlliedWare Plus firewalls and routers

show pbr rules

This command displays a list of the rules that have been configured:

```
AE3050S#show pbr rules
Policy based routing is enabled

Rule Match   From       To                 Valid       Nexthop
---------------------------------------------------------------------
30   tcp1      any      entities.outside    Yes        172.16.2.2
                                                       172.16.3.2
                                                       172.16.4.2
                                                       172.16.5.2
40   tcp2      any      entities.outside    Yes        172.16.7.2
                                                       172.16.8.2
                                                       172.16.9.2
                                                       172.16.10.2
```

show ip pbr route <*1-255*>

This will show the routes that are created by the rules. You can also narrow the display by specifying the PBR rule ID. The output shows the main IP route table (the standard tale that does not contain policy routes), followed by the sub-tables created for the policy rules. Each rule creates it own sub-table.

```
awplus#show ip pbr route
Route table: main
    10.10.10.0/30 is directly connected, vlan10
    10.10.20.0/30 is directly connected, vlan20
    10.10.30.0/30 via 10.10.10.2, vlan10
    10.10.100.0/24 via 10.10.10.2, vlan10
    10.37.236.32/27 is directly connected, vlan100
    172.16.2.0/30 is directly connected, vlan200
    172.16.3.0/24 is directly connected, vlan201
    172.16.4.0/24 is directly connected, vlan202
    172.16.5.0/24 is directly connected, vlan203
    172.31.0.0/17 is directly connected, vlan4092

Route table: policy-route 30
    default via 172.16.2.2, vlan200
    default via 172.16.3.2, vlan201
    default via 172.16.4.2, vlan202
    default via 172.16.5.2, vlan203

Route table: policy-route 40
    default via 10.10.20.2, vlan20

Route table: policy-route 50
    default is directly connected, ppp1
    default is directly connected, tunnel10
```

Similarly the command **show ipv6 pbr route** outputs the main IPv6 route table, and the policy-related sub-tables.

```
awplus#show ipv6 pbr route
Route table: main
    2001:100::/64 dev vlan100
    fe80::/64 dev eth1
    fe80::/64 dev vlan100

Route table: policy-route 20
    default via 2001:100::2, vlan100
```

Debugging

The command **debug policy-based-routing** enables a debug that will output a message each time a rule is matched.

```
AR3050S#23:38:42 AR3050S kernel: PBR: IN=vlan100-4-1 OUT=
MAC=00:00:5e:00:01:01:00:1a:eb:92:42:6e:08:00 SRC=192.168.100.3 DST=1.1.1.1
LEN=38 TOS=0x00 PREC=0x00 TTL=1 ID=28200
23:38:42 AR3050S kernel: PBR: IN=vlan100-4-1 OUT=
MAC=00:00:5e:00:01:01:00:1a:eb:92:42:6e:08:00 SRC=192.168.100.3 DST=1.1.1.1
LEN=38 TOS=0x00 PREC=0x00 TTL=1 ID=28201 DF PROTO=
23:38:42 AR3050S kernel: PBR: IN=vlan100-4-1 OUT=
MAC=00:00:5e:00:01:01:00:1a:eb:92:42:6e:08:00 SRC=192.168.100.3 DST=1.1.1.1
LEN=38 TOS=0x00 PREC=0x00 TTL=1 ID=28202 DF PROTO=
23:38:42 AR3050S kernel: PBR: IN=vlan100-4-1 OUT=
MAC=00:00:5e:00:01:01:00:1a:eb:92:42:6e:08:00 SRC=192.168.100.3 DST=1.1.1.1
LEN=38 TOS=0x00 PREC=0x00 TTL=2 ID=28203 DF PROTO=
23:38:42 AR3050S kernel: PBR: IN=vlan100-4-1 OUT=
MAC=00:00:5e:00:01:01:00:1a:eb:92:42:6e:08:00 SRC=192.168.100.3 DST=1.1.1.1
LEN=38 TOS=0x00 PREC=0x00 TTL=2 ID=28204 DF PROTO=
23:38:42 AR3050S kernel: PBR: IN=vlan100-4-1 OUT=
MAC=00:00:5e:00:01:01:00:1a:eb:92:42:6e:08:00 SRC=192.168.100.3 DST=1.1.1.1
LEN=38 TOS=0x00 PREC=0x00 TTL=2 ID=28205 DF PROTO=
23:38:42 AR3050S kernel: PBR: IN=vlan100-4-1 OUT=
MAC=00:00:5e:00:01:01:00:1a:eb:92:42:6e:08:00 SRC=192.168.100.3 DST=1.1.1.1
LEN=38 TOS=0x00 PREC=0x00 TTL=3 ID=28206 DF PROTO=
23:38:42 AR3050S kernel: PBR: IN=vlan100-4-1 OUT=
MAC=00:00:5e:00:01:01:00:1a:eb:92:42:6e:08:00 SRC=192.168.100.3 DST=1.1.1.1
LEN=38 TOS=0x00 PREC=0x00 TTL=3 ID=28207 DF PROTO=
23:38:42 AR3050S kernel: PBR: IN=vlan100-4-1 OUT=
MAC=00:00:5e:00:01:01:00:1a:eb:92:42:6e:08:00 SRC=192.168.100.3 DST=1.1.1.1
LEN=38 TOS=0x00 PREC=0x00 TTL=3 ID=28208 DF PROTO=
```

The debug will also output detailed information when you attempt to create an invalid rule:

```
23:51:42 AR3050S IMISH[24833]: ip policy-route match test1 from test2 to test3
nexthop 5.5.5.5

AR3050S(config-pbr)#23:51:42 AR3050S pbrd: PBR: Can't find application test1
23:51:42 AR3050S PBR: --add-route 4 40 -m set --match-set test2 src,src -m set -
-match-set test3 dst,dst -4 5.5.5.5
23:51:42 AR3050S PBR: ipt -4 -N PBR_RULE_40
23:51:42 AR3050S PBR: ipt -4 -F PBR_RULE_40
23:51:42 AR3050S PBR: ipt -4 -A PBR_RULE_40 -m set --match-set test2 src,src -m
set --match-set test3 dst,dst -m conntrack --ctstate NEW,RELATED,ESTABLISHED -j
CONNMARK --set-m0
23:51:42 AR3050S PBR: ipt -4 -N PBR_RULES
23:51:42 AR3050S PBR: ipt -4 -D PBR_RULES -m connmark --mark 0x0/0xff000000 -j
PBR_RULE_40
23:51:42 AR3050S PBR: ipt -4 -I PBR_RULES 4 -m connmark --mark 0x0/0xff000000 -j
PBR_RULE_40
23:51:42 AR3050S PBR: reset connmark --mark 0/0xff000000
23:51:42 AR3050S PBR: -4 rule add fwmark 0x28000000/0xff000000 pref 1024 table 2040
23:51:42 AR3050S PBR: -4 route append default via 5.5.5.5 table 2040
23:51:42 AR3050S pbrd: PBR: Failed to add policy route 40
```

The debug outputs information when an interface state change results in an active next-hop being added to or deleted from a rule.

```
00:05:30 AR3050S kernel: IPv6: ADDRCONF(NETDEV_CHANGE): eth2: link becomes ready
00:05:30 AR3050S NSM[565]: Port up notification received for eth2

AR3050S(config-if)#00:05:30 AR3050S PBR: --add-route 1 10 -p 17 -m multiport --
sports 1024:65535 -m multiport --dports 33434:33523 -m set --match-set
entities.any src,src -m s6
00:05:30 AR3050S PBR: ipt -4 -N PBR_RULE_10
00:05:30 AR3050S PBR: ipt -4 -F PBR_RULE_10
00:05:31 AR3050S PBR: ipt -4 -A PBR_RULE_10 -p 17 -m multiport --sports 1024:65535
-m multiport --dports 33434:33523 -m set --match-set entities.any src,src -m set
--match-set 0
00:05:31 AR3050S PBR: ipt -4 -N PBR_RULES
00:05:31 AR3050S PBR: ipt -4 -D PBR_RULES -m connmark --mark 0x0/0xff000000 -j
PBR_RULE_10
00:05:31 AR3050S PBR: ipt -4 -I PBR_RULES 1 -m connmark --mark 0x0/0xff000000 -j
PBR_RULE_10
00:05:31 AR3050S PBR: reset connmark --mark 0/0xff000000
00:05:31 AR3050S PBR: -4 rule add fwmark 0x0a000000/0xff000000 pref 1024 table 2010
00:05:31 AR3050S PBR: -4 route append default via 172.16.1.2 table 2010
00:05:31 AR3050S PBR: -4 route append default via 10.10.10.6 table 2010
00:05:31 AR3050S PBR: --add-route 2 20 -p 17 -m multiport --sports 1024:65535 -m
multiport --dports 33434:33523 -m set --match-set entities.outside dst,dst -4
172.16.1.1
00:05:31 AR3050S PBR: ipt -4 -N PBR_RULE_20
00:05:31 AR3050S PBR: ipt -4 -F PBR_RULE_20
00:05:31 AR3050S PBR: ipt -4 -A PBR_RULE_20 -p 17 -m multiport --sports 1024:65535
-m multiport --dports 33434:33523 -m set --match-set entities.outside dst,dst -m
conntrack --0
00:05:31 AR3050S PBR: ipt -4 -N PBR_RULES
00:05:31 AR3050S PBR: ipt -4 -D PBR_RULES -m connmark --mark 0x0/0xff000000 -j
PBR_RULE_20
00:05:31 AR3050S PBR: ipt -4 -I PBR_RULES 2 -m connmark --mark 0x0/0xff000000 -j
PBR_RULE_20
00:05:31 AR3050S PBR: reset connmark --mark 0/0xff000000
00:05:31 AR3050S PBR: -4 rule add fwmark 0x14000000/0xff000000 pref 1024 table 2020
00:05:31 AR3050S PBR: -4 route append default via 172.16.1.1 table 2020
```

The command **show debugging policy-based-routing** shows if debugging is currently enabled or disabled:

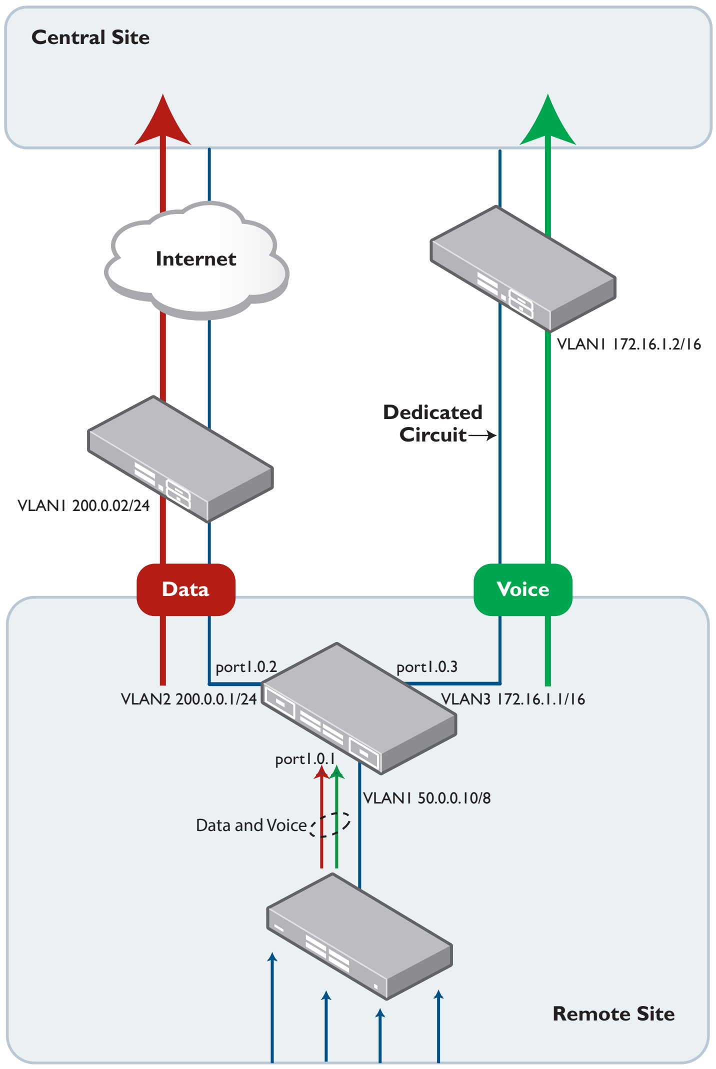
```
AR3050S#show debugging policy-based-routing

Policy Based Routing Debugging Status: on
```

Configuration example 1: managing VoIP using PBR

VoIP is being used to provide voice communications within an organization. To ensure high voice quality, dedicated data circuits are leased between remote sites, and a central site. All VoIP data is routed via these dedicated circuits from the remote sites to a central site. The VoIP is then distributed from the central site to its eventual destinations. By using this hub-and-spoke arrangement of low-bandwidth circuits for transporting VoIP, the organization is provided with good-quality voice communications in a cost-effective manner. All other data communication between the sites is transported over the Internet. The VoIP traffic is marked with a DSCP value of 46.

Policy routing is used to ensure that the VoIP packets are sent via the dedicated circuit, whilst all other data is sent over the Internet. The VoIP traffic is also put into the highest queue on the egress port, to ensure minimum packet loss and delay.



Configuration if using a switch

Step 1: Create the VLANs and assign IP addresses to them.

```
awplus(config)# vlan database
awplus(config-vlan)# vlan 2-3 state enable
```

VLAN1 connects to the local LAN:

```
awplus(config)# interface vlan1
awplus(config-if)# ip address 50.0.0.10/8
```

VLAN2 connects to the Internet:

```
awplus(config)# interface vlan2
awplus(config-if)# ip address 200.0.0.1/30
```

VLAN3 is directed towards the dedicated circuit:

```
awplus(config)# interface vlan3
awplus(config-if)# ip address 172.16.1.1/30
```

Step 2: **Assign the VLANs to the ports**

Port1.0.1 is already in VLAN 1 by default:

```
awplus(config)# interface port1.0.1
awplus(config-if)# switchport
awplus(config-if)# switchport mode access
```

Assign port1.0.2 to VLAN2:

```
awplus(config)# interface port1.0.2
awplus(config-if)# switchport
awplus(config-if)# switchport mode access
awplus(config-if)# switchport access vlan2
```

Assign port1.0.2 to VLAN3:

```
awplus(config)#interface port1.0.3
awplus(config-if)# switchport
awplus(config-if)# switchport mode access
awplus(config-if)# switchport access vlan3
```

Step 3: **Configure the default route. This is the route to the Internet**

```
awplus(config)# ip route 0.0.0.0/0 200.0.0.2
```

Step 4: **Enable QoS globally**

```
awplus(config)# mls qos enable
```

Step 5: **Create the class-map that will match on traffic with a DSCP value of 46**

```
awplus(config)# class-map test
awplus(config-cmap)# match dscp 46
```

Step 6: Create the policy-map that will specify the actions to be taken on the classified traffic

The first action instructs the switch to send this packet to queue 7 on the egress port. The second action sets the next-hop for this traffic to 172.16.1.2, so that it will be directed towards the dedicated circuit.

```
awplus(config)# policy-map pbr
awplus(config-pmap)# class test
awplus(config-pmap-c)# set queue 7
awplus(config-pmap-c)# set ip next-hop 172.16.1.2
```

Step 7: Finally, attach the policy-map to the ingress port

```
awplus(config)# interface port1.0.1
awplus(config-if)# service-policy input pbr
```

Configuration if using a firewall or router

Step 1: Assign IP addresses to interfaces

```
awplus(config)# interface vlan1
awplus(config-if)# ip address 50.0.0.10/8
```

eth1 connects to the Internet

```
awplus(config)# interface eth1
awplus(config-if)# ip address 200.0.0.1/30
```

eth2 is directed towards the dedicated circuit

```
awplus(config)# interface eth2
awplus(config-if)# ip address 172.16.1.1/30
```

Step 2: Configure the default route. This is the route to the Internet

```
awplus(config)# ip route 0.0.0.0/0 200.0.0.0
```

Step 3: Create the application that will match on traffic with a DSCP value of 46

```
awplus(config)# application voice
awplus(config-application)# protocol udp
awplus(config-application)# dscp 46
```

Step 4: Create zones define the sources and destinations of the traffic

```
awplus(config)# zone private
awplus(config-zone)# network internal
awplus(config-zone-network)# ip subnet 50.0.0.10/8
awplus(config)# zone WAN
awplus(config-zone)# network dedicated
awplus(config-network)# ip subnet 172.16.1.1/30
awplus(config)# zone internet
awplus(config-zone)# network public
awplus(config-network)# ip subnet 0.0.0.0/0
```

Step 5: **Enable policy-based routing**

```
awplus(config)# policy-based-routing
awplus(config-PBR)# policy-based-routing enable
```

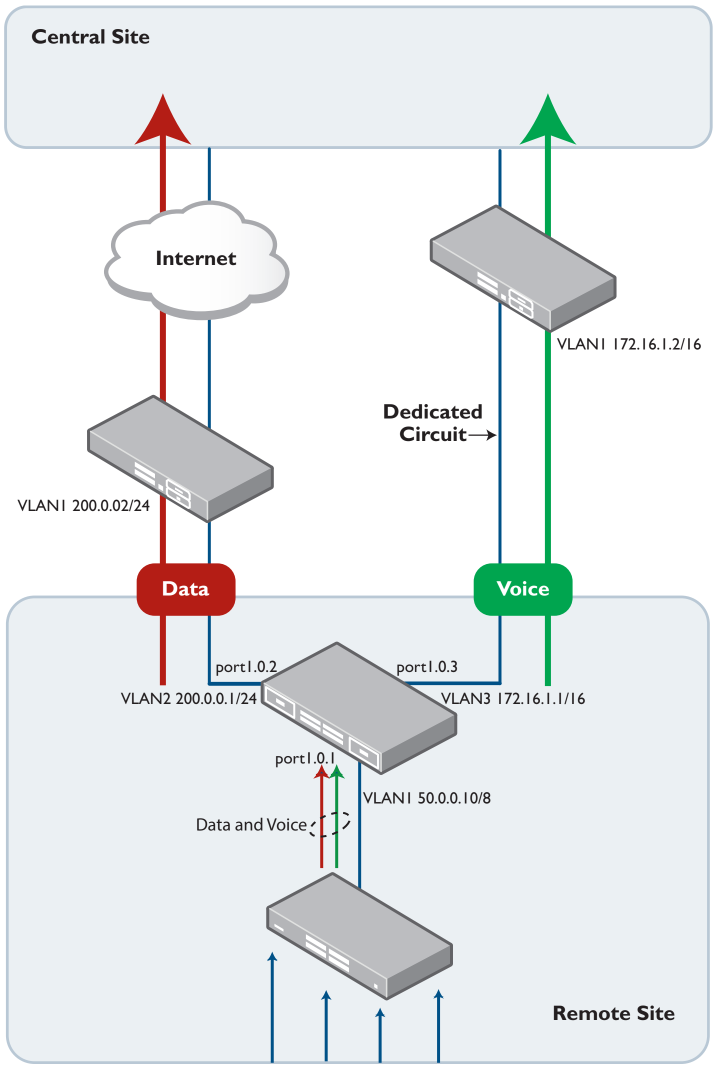
Step 6: **Add the rule to direct the voice traffic via the dedicated link**

```
awplus(config-PBR)# ip policy-route 10 voice from private nexthop
172.16.1.2
```

Configuration example 2: managing VoIP using application-aware PBR

VoIP is being used to provide voice communications within an organization. To ensure high voice quality, dedicated data circuits are leased between remote sites, and a central site. All VoIP data is routed via these dedicated circuits from the remote sites to a central site. The VoIP is then distributed from the central site to its eventual destinations. By using this hub-and-spoke arrangement of low-bandwidth circuits for transporting VoIP, the organization is provided with good-quality voice communications in a cost-effective manner. All other data communication between the sites is transported over the Internet. The VoIP traffic is marked with a DSCP value of 46.

Policy routing is used to ensure that the VoIP packets are sent via the dedicated circuit, whilst all other data is sent over the Internet. The VoIP traffic is also put into the highest queue on the egress port, to ensure minimum packet loss and delay.



Configuration if using a firewall or router

Step 1: Assign IP addresses to interfaces

```
awplus(config)# interface vlan1
awplus(config-if)# ip address 50.0.0.10/8
```

eth1 connects to the Internet:

```
awplus(config)# interface eth1
awplus(config-if)# ip address 200.0.0.1/30
```

eth2 is directed towards the dedicated circuit:

```
awplus(config)# interface eth2
awplus(config-if)# ip address 172.16.1.1/30
```

Step 2: Configure the default route. This is the route to the Internet

```
awplus(config)# ip route 0.0.0.0/0 200.0.0.0
```

Step 3: Create the linkmon probe for Skype traffic

```
awplus(config)# linkmon probe name SkypeProbe type ping destination
172.16.1.1/30
```

Step 4: Create the linkmon group for Skype users

```
awplus(config)# linkmon group SkypeGroup
awplus(config)-linkmon-group)# member 1 destination 172.16.1.1/30 probe
SkypeProbe
awplus(config)-linkmon-group)# exit
```

Step 5: Create the linkmon profile

```
awplus(config)# linkmon profile SkypeProfile
awplus(config)-linkmon-profile)# jitter bad-above 200
awplus(config)-linkmon-profile)# latency bad-above 200
awplus(config)-linkmon-profile)# pktloss bad-above 5
awplus(config)-linkmon-profile)# preference jitter
awplus(config)-linkmon-profile)# exit
```

Step 6: Create zones define the sources and destinations of the traffic

```
awplus(config)# zone private
awplus(config-zone)# network internal
awplus(config-zone-network)# ip subnet 50.0.0.10/8
awplus(config)# zone WAN
awplus(config-zone)# network dedicated
awplus(config-network)# ip subnet 172.16.1.1/30
awplus(config)# zone internet
awplus(config-zone)# network public
awplus(config-network)# ip subnet 0.0.0.0/0
```

Step 7: Enable policy-based routing

```
awplus(config)# policy-based-routing
```

```
awplus(config-PBR)# policy-based-routing enable
```

Step 8: Add the rule to direct Skype traffic via the dedicated link

```
awplus(config-PBR)# ip policy-route 10 linkmon-group SkypeGroup linkmon-
profile SkypeProfile
```

Configuration example 3: reducing data transmission costs

In this scenario (using the same diagram as in example 1 above) we have a faster Internet connection on port1.0.2 (if using a switch, or eth1 if using a firewall). This connection costs the company more to use. The ISP providing this faster connection charges on the basis of the amount of data sent over the connection.

The company has decided that traffic from their web server will be sent to the Internet via this connection on Monday to Friday during business hours only (9.00am to 5.30 pm). This provides good web service during business hours, whilst keeping some limit on the total amount of data sent over the faster (more expensive) connection. Outside these times, traffic from the web server is sent via the default route. Any other traffic is always sent via the default route.

Additionally a ping poll can be configured that will regularly check that the Policy Route next-hop is reachable. If the ping poll fails to get a response from the next-hop, a trigger will be run that removes the policy from the ingress port. This will ensure that if the Policy Route next-hop fails, the configured default route will be used, ensuring no loss of connectivity. Once the next-hop is reachable again, another trigger adds the policy back onto the ingress port.

Configuration if using a switch

Step 1: **Create the VLANs and assign IP addresses to them**

```
awplus(config)# vlan database
awplus(config-vlan)# vlan 2-3 state enable
```

Assign an IP address to VLAN1:

```
awplus(config)# interface vlan1
awplus(config-if)# ip address 50.0.0.10/8
```

Assign an IP address to VLAN2:

```
awplus(config)# interface vlan2
awplus(config-if)# ip address 200.0.0.1/30
```

Assign an IP address to VLAN3:

```
awplus(config)# interface vlan3
awplus(config-if)# ip address 172.16.1.1/30
```

Step 2: **Assign the VLANs to the ports**

Port 1.0.1 is already in VLAN1 by default:

```
awplus(config)# interface port1.0.1
awplus(config-if)# switchport
awplus(config-if)# switchport mode access
```

Assign port1.0.2 to VLAN2:

```
awplus(config)# interface port1.0.2
awplus(config-if)# switchport
awplus(config-if)# switchport mode access
awplus(config-if)# switchport access vlan 2
```

Assign port1.0.3 to VLAN3:

```
awplus(config)# interface port1.0.3
awplus(config-if)# switchport
awplus(config-if)# switchport mode access
awplus(config-if)# switchport access vlan 3
```

Step 3: **Configure the default route**

```
awplus(config)# ip route 0.0.0.0/0 172.16.1.2
```

Step 4: **Enable QoS globally**

```
awplus(config)# mls qos enable
```

Step 5: **Create the access-list that classifies on traffic from the Web server**

```
awplus(config)# access-list 3001 permit tcp <web-server-IP address> eq 80
any
```

Step 6: **Apply this access-list to a class-map**

```
awplus(config)# class-map web-server
awplus(config-cmap)# match access-group 3001
```

Step 7: **Apply this class-map to a Policy-map and configure the policy to send the traffic matching class web-server to a specific next-hop**

```
awplus(config)# policy-map pbr
awplus(config-pmap)# class web-server
awplus(config-pmap-c)# set ip next-hop 200.0.0.2
```

Step 8: **Create the script that will add the policy to the ingress port when trigger 1 is run**

```
Edit policy-on.scp
enable
conf t
int port1.0.1
service-policy input pbr
```

Step 9: Create the script that will remove the policy from the ingress port when trigger 2 is run

```
Edit policy-off.scp

enable

conf t

int port1.0.1

no service-policy input pbr
```

Step 10: Configure the trigger that will add the policy to the ingress port at the required day/time

```
awplus(config)# trigger 1

awplus(config-trigger)# type time 09:00

awplus(config-trigger)# day monday tuesday wednesday thursday friday

awplus(config-trigger)# script 1 policy-on.scp
```

Step 11: Configure the trigger that will remove the policy from the ingress port at the required day/time

```
awplus(config)# trigger 2

awplus(config-trigger)# type time 17:30

awplus(config-trigger)# day monday tuesday wednesday thursday friday

awplus(config-trigger)# script 1 policy-off.scp
```

Step 12: Configure the ping poll that will regularly check that the next-hop is reachable

```
awplus(config)# ping-poll 1

awplus(config-ping-poll)# description “check policy route next hop”

awplus(config-ping-poll)# ip 200.0.0.2

awplus(config-ping-poll)# source-ip 200.0.0.1

awplus(config-ping-poll)# active
```

Step 13: Configure the trigger that will be activated when the ping poll fails

```
awplus(config)# trigger 3

awplus(config-trigger)# type ping-poll 1 down

awplus(config-trigger)# script 1 policy-off.scp

awplus(config-trigger)# active

awplus(config-trigger)# time after 09:00:00 before 17:30:00

awplus(config-trigger)# day monday tuesday wednesday thursday friday
```

Step 14: Configure the trigger that will be activated when the next-hop is reachable

```
awplus(config)# trigger 4

awplus(config-trigger)# type ping-poll 1 up

awplus(config-trigger)# script 1 policy-on.scp

awplus(config-trigger)# active

awplus(config-trigger)# time after 09:00:00 before 17:30:00
awplus(config-trigger)# day monday tuesday wednesday thursday friday
```

Configuration if using a firewall or router

Step 1: Assign IP addresses to interfaces

VLAN1 connects to the local LAN:

```
awplus(config)# interface vlan1
awplus(config-if)# ip address 50.0.0.10/8
```

eth1 connects to the Internet:

```
awplus(config)# interface eth1
awplus(config-if)# ip address 200.0.0.1/30
```

eth2 is directed towards the dedicated circuit:

```
awplus(config)# interface eth2
awplus(config-if)# ip address 172.16.1.1/30
```

Step 2: Configure the default route. This is the route to the Internet

```
    awplus(config)# ip route 0.0.0.0/0 200.0.0.0
```

Step 3: Create zones to define the sources and destinations of the traffic

Create a private zone:

```
awplus(config)# zone private
awplus(config-zone)# network internal
awplus(config-network)# ip subnet 50.0.0.10/8
awplus(config-network)# host webserver
awplus(config-host)# ip address <server address>
```

Create an internet zone:

```
awplus(config)# zone internet
awplus(config-zone)# network public
awplus(config-network)# ip subnet 0.0.0.0/0
```

Step 4: Enable Policy-based routing

```
awplus(config)# policy-based-routing
awplus(config-PBR)# policy-based-routing enable
```

Step 5: Add the rule to direct the voice traffic via the dedicated link

```
awplus(config-PBR)# ip policy-route 10 http from
private.internal.webserver nexthop 200.0.0.2
```

Step 6: Create the script that will enable PBR when trigger 1 is run

```
Edit policy-on.scp
enable
conf t
policy-based-routing
policy-based-routing enable
```

Step 7: Create the script that will disable PBR when trigger 2 is run

```
Edit policy-on.scp
enable
conf t
policy-based-routing
no policy-based-routing enable
```

Step 8: Configure the trigger that will add the policy to the ingress port at the required day/time

```
awplus(config)# trigger 1
awplus(config-trigger)# type time 09:00
awplus(config-trigger)# day monday tuesday wednesday thursday friday
awplus(config-trigger)# script 1 policy-on.scp
```

Step 9: Configure the trigger that will remove the policy from the ingress port at the required day/time

```
awplus(config)# trigger 2
awplus(config-trigger)# type time 17:30
awplus(config-trigger)# day monday tuesday wednesday thursday friday
awplus(config-trigger)# script 1 policy-off.scp
```

Step 10: Configure the ping poll that will regularly check that the next-hop is reachable

```
awplus(config)# ping-poll 1
awplus(config-ping-poll)# description “check policy route next hop”
awplus(config-ping-poll)# ip 200.0.0.2
awplus(config-ping-poll)# source-ip 200.0.0.1
awplus(config-ping-poll)# active
```

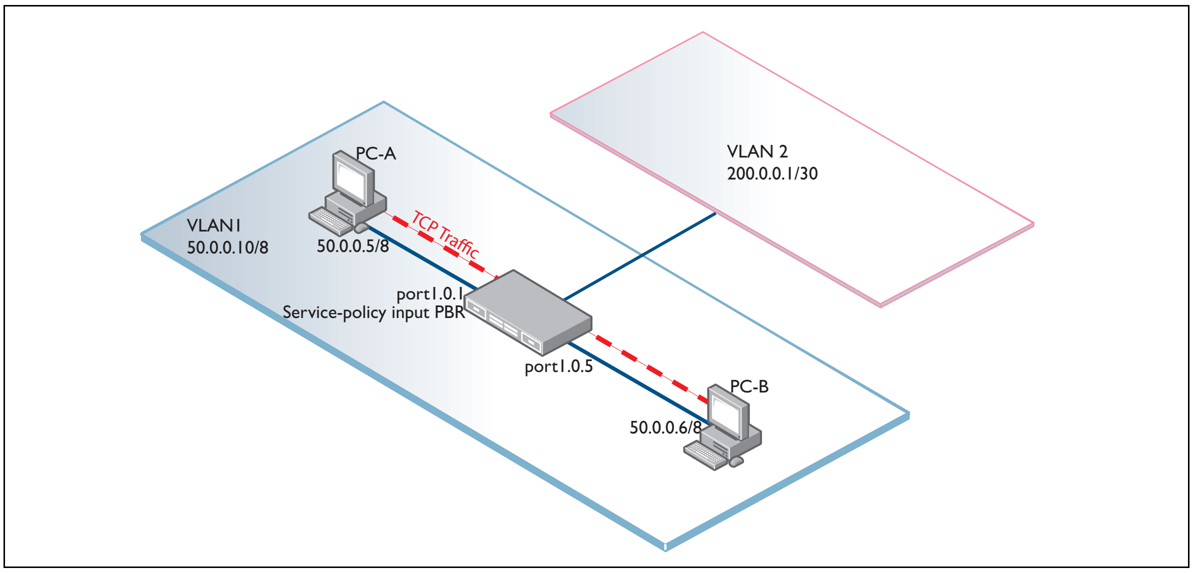
Step 11: Configure the trigger that will be activated when the ping poll fails

```
awplus(config)# trigger 3
awplus(config-trigger)# type ping-poll 1 down
awplus(config-trigger)# script 1 policy-off.scp
awplus(config-trigger)# active
awplus(config-trigger)# time after 09:00:00 before 17:30:00
awplus(config-trigger)# day monday tuesday wednesday thursday friday
```

Step 12: Configure the trigger that will be activated when the next-hop is reachable

```
awplus(config)# trigger 4
awplus(config-trigger)# type ping-poll 1 up
awplus(config-trigger)# script 1 policy-on.scp
awplus(config-trigger)# active
awplus(config-trigger)# time after 09:00:00 before 17:30:00
awplus(config-trigger)# day monday tuesday wednesday thursday friday
```

Example 4: routing TCP traffic



When using policy-based routing, there is a distinct difference between the behavior of the switches and that of the firewalls and routers with regard to traffic destined to the device itself. In the case of the switches, traffic that's destined for the switch itself can be unexpectedly routed to the PBR next-hop. This includes TCP traffic such as telnet. In the example below, all TCP traffic is to be Policy Routed.In the case of a switch, this will also affect Telnet to the switch itself. To avoid this, extra configuration is required.

In the case of the firewalls and routers, the traffic destined to the device itself is not subject to Policy Based Routing, so no extra configuration is required to avoid that occurring.

Configuration if using a switch

Step 1: **Create two IP interfaces: VLAN1 and VLAN2**

Give VLAN1 an IP address:

```
awplus(config)# interface vlan1
awplus(config-if)# ip address 50.0.0.10/8
```

Give VLAN2 an IP address:

```
awplus(config)# interface vlan2
awplus(config-if)# ip address 200.0.0.1/30
```

We do not want traffic destined to the addresses of either of these interfaces to be Policy Routed.

Step 2: Create ACLs for any TCP traffic destined for these networks, as well as the PBR ACL

```
awplus(config)# mls qos enable
awplus(config)# access-list 3001 permit tcp any 50.0.0.0/8
awplus(config)# access-list 3002 permit tcp any 200.0.0.1/32
awplus(config)# access-list 3003 permit tcp any any
```

In the diagram above, any traffic matching the Policy-based routing access-list 3003, which classifies on TCP traffic ingressing port1.0.1, will be sent to the next-hop address of 200.0.0.2 on VLAN2. Even L2 traffic in the same VLAN through the switch will be sent to the PBR next-hop if it matches the PBR access-list.

This would mean that any TCP traffic from PC-A destined for another device connected to the switch in the same VLAN (e.g. PC-B) would not be forwarded to its destination but sent to the next-hop of 200.0.0.2. So, we configure ACL 3001 to match on any TCP traffic destined for the entire subnet in use on VLAN1, not just the IP address of the switch itself.

Allied Telesis recommends that the PBR policy always includes a class to allow traffic. This class must precede the Policy Routing class.

Step 3: Apply these access-lists to class-maps

Class-map local1:

```
awplus(config)# class-map local1
awplus(config-cmap)# match access-group 3001
```

Class-map local2:

```
awplus(config)# class-map local2
awplus(config-cmap)# match access-group 3002
```

Class-map local3:

```
awplus(config)# class-map tcp
awplus(config-cmap)# match access-group 3003
```

Step 4: Apply the class-maps to a Policy-map and configure the policy to send the matching TCP traffic to a specific next-hop

The effect of the policy map is that:

- Any traffic matching classes local1 or local2 (i.e. ACLs 3001 or 3002) will simply pass through to the normal forwarding process.
- Any traffic matching class tcp will be policy routed.

```
awplus(config)# policy-map pbr
awplus(config-pmap)# class local1
awplus(config-pmap)# class local2
awplus(config-pmap)# class tcp
awplus(config-pmap-c)# set ip next-hop 200.0.0.2
```

Step 5: Finally, attach the policy-map to the ingress port

```
awplus(config)# interface port1.0.1
awplus(config-if)# switchport
awplus(config-if)# switchport mode access
awplus(config-if)# service-policy input pbr
```

More detail of the configuration of policy-based routing can be seen in the QoS chapter of the switch's software reference manuals.

Configuration if using a firewall or router

Step 1: Create the zone for the public side of the firewall

```
awplus(config)# zone internet
awplus(config-zone)# network public
awplus(config-network)# ip subnet 0.0.0.0/0
awplus(config-network)# host RouterA
awplus(config-host)# ip address 200.0.0.1
awplus(config-host)# host RouterB
awplus(config-host)# ip address 200.0.0.4
```

Step 2: Create the zone for the private side of the firewall

```
awplus(config)# zone private
awplus(config-zone)# network internal
awplus(config-network)# ip subnet 50.0.0.0/8
awplus(config-network)# host A
awplus(config-host)# ip address 50.0.0.5
awplus(config-host)# host B
awplus(config-host)# ip address 50.0.0.6
```

Step 3: Apply IP addresses to interfaces

Give eth1 an IP address:

```
awplus(config)# interface eth1
awplus(config-if)# ip address 200.0.0.1/30
```

Give VLAN1 an IP address:

```
awplus(config)# interface vlan1
awplus(config-if)# ip address 50.0.0.10/8
```

Step 4: Create the default route to the Internet

```
awplus(config)# ip route 0.0.0.0/0 200.0.0.4
```

Step 5: Set up policy based routing

```
awplus(config)# policy-based-routing
```

```
awplus(config-PBR)# policy-based-routing enable
```

```
awplus(config-PBR)# ip policy-route 20 match TCP from private to internet
nexthop 200.0.0.2
```

If PBR debug is enabled on the firewall, then the packets in a TCP session from a host on the private LAN, destined to the Internet, will show up in the debug, like this:

```
awplus#09:26:39 awplus kernel: PBR: IN=vlan1
OUT=MAC=02:00:00:56:78:9c:ec:cd:6d:82:6c:7f:08:00 SRC=50.0.0.6 DST=200.0.0.4
LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=4799 DF PROTO=TCP SPT=41272 DPT=23
WINDOW=14600 RES=0x00 SYN URGP=0 MARK=0xff000000
```