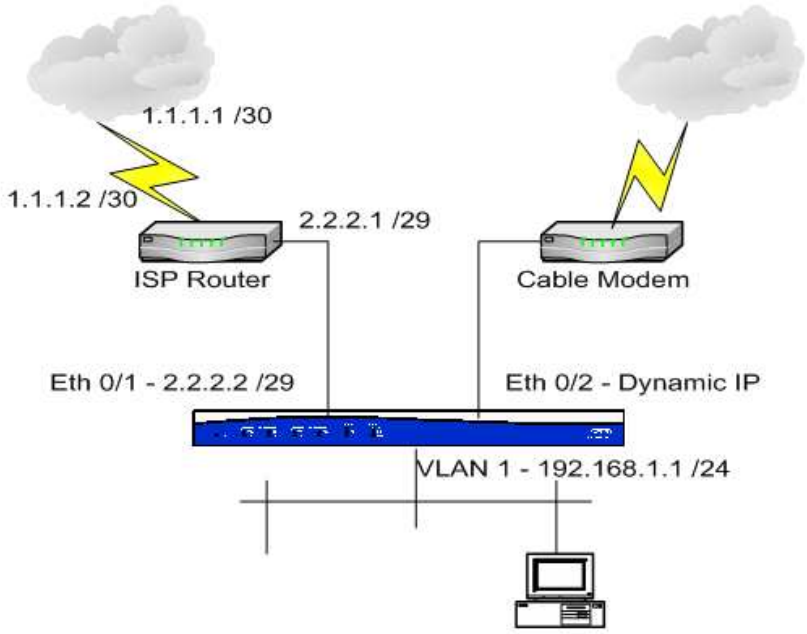
March 2010

Quick Configuration Guide
Configuring WAN Fail-Over in AOS

Configuring WAN Fail-Over in AOS



Introduction

To minimize downtime, many businesses are moving to multiple Wide-Area Network (WAN) connections. This article will help explain how to properly implement the second WAN connection, and fail-over for any possible scenario. This document will not cover using both WAN connections simultaneously, although the processes learned through this document will cover the majority of the required configuration for load-sharing. Load-sharing is specifically covered in KB article #1994.

Hardware/Software Requirements

WAN fail-over requires that the following criteria to be met:

- If any Internet connection is through an Ethernet hand-off, the device must support Network Monitor, which includes the NetVanta 1335, NetVanta 3000 series (except for the 3200/3205), NetVanta 4000 series, and NetVanta 5000 series.

- If any Internet connection is through an Ethernet hand-off, the device needs to run AOS 13.1 or higher

Overview

With WAN fail-over, there are several factors that must be configured to allow for the fail-over process to occur properly. They include:

- Firewall Settings
- Routing Settings
- Dynamic Interface IP Settings (if applicable)
- Interface Tracking (if applicable)

Firewall Settings

Regardless of the type of connections being used, the firewall structure is always the same.

The primary issue with multiple WAN connections is that each connection must have a uniquely identified policy-class. The policy-classes can be identically configured, but must be separate. This is because the firewall needs to have a mechanism to differentiate between interfaces.

In the private side policy-class(es), multiple NAT statements must be defined, specifying the destination policy-class. Each NAT statement should use a different access-list as reference, even though in most cases they will be identical. The key is not to use the same access-list multiple times in the same policy-class, as it can sometimes trigger instability, especially if the firewall is later edited through the web interface.

The firewall needs to be placed in 'fast-nat-failover mode', which will dynamically clear all current policy-sessions when a change in the default route occurs. This is important because if the sessions were not cleared, the router would continue to send traffic out to the interface that is down. Normal service would not return until the sessions time-out, which could take as much as twenty (20) minutes by default.

A sample config is shown here:

```
ip firewall
ip firewall fast-nat-failover
!
ip access-list extended AdminAccess
  permit tcp any  any eq www   log
  permit tcp any  any eq telnet   log
  permit udp any  any eq snmp    log
  permit tcp any  any eq https   log
  permit tcp any  any eq ssh   log
  permit tcp any  any eq ftp   log
  permit icmp any  any  echo   log
```

```
!
ip access-list extended NAT-Primary
  permit ip any any
!
ip access-list extended NAT-Backup
  permit ip any any
!
ip policy-class WAN1
  allow list AdminAccess self
!
ip policy-class WAN2
  allow list AdminAccess self
!
ip policy-class Private
  nat source list NAT-Primary interface <WAN1 interface> overload
     policy WAN1
  nat source list NAT-Backup interface <WAN2 interface> overload
     policy WAN2
```

NOTE: *<WAN1 interface> and <WAN2 interface> should be replaced with the appropriate interfaces on the router being configured (ppp 1, vlan 1, etc).*

Routing Settings

In a primary and backup WAN configuration, there needs to be a mechanism to allow the router to know which route is the most preferable. This is accomplished by the administrative distance property.

The administrative distance property specifies in what order routes should be added to the route table. It is processed from lowest to highest. A route with no administrative distance adopts the default administrative distance of one (1). All secondary routes should then have distances higher than the previous route when they are added to the route table.

In the simplest form of primary and backup WAN configurations, the routes are added statically. The command syntax for adding a static route is:

ip route *<Destination IP Address> <Destination Mask> <Next-Hop IP Address or Egress Interface> <Administrative Distance>*

In the case of an Internet connection, the destination address and mask consist of all zeros (0s) so it will match all traffic, making it the 'default' route when a more specific route is not available. A sample configuration of two default routes is shown below. Notice the administrative distance of 10 added to the second route at the end of the command:

```
ip route 0.0.0.0 0.0.0.0 <WAN1 Next-Hop IP Address OR WAN1 Interface>
!
ip route 0.0.0.0 0.0.0.0 <WAN2 Next-Hop IP Address OR WAN2 Interface> 10
```

The next question is how to add an administrative distance to routes that are learned dynamically. This depends on the type of Internet connection being used and will be covered in the next section of this article.

Dynamic Interface IP Settings (if applicable)

If an IP interfaces on the router is obtaining its address dynamically, there is no way to predict the gateway/peer IP address that all traffic needs to be routed towards. This section describes how to create a proper routing table structure using a PPP interface with a negotiated address, or an Ethernet address using DHCP.

On a PPP interface, the **ip address negotiated** command will tell the router to obtain its IP address from the PPP peer. The router will then apply this address to itself and add the IP address of the peer as its default route with the default administrative distance of one (1). If the PPP connection is the primary connection, this requires no re-configuration. If the PPP connection is the secondary connection, an extra option must be added to the configuration, which is only allowable through the command line interface (CLI). The command syntax is:

ip address negotiated <*administrative distance*>

A sample configuration is shown here:

```
interface ppp 1
  ip address negotiated
!
interface ppp 2
  ip address negotiated 10
```

In the case of an Ethernet connection gaining its address via DHCP, the circumstances that necessitated the use of an administrative distance on the PPP interface apply here as well. The command syntax is:

ip address dhcp <*administrative distance*>

A sample configuration is shown here:

```
interface eth 0/1
  ip address dhcp
!
interface eth 0/2
  ip address dhcp 10
```

Routing Fail-Over

The final question then becomes how the router knows to 'pull' the primary route out when it is not needed, and 'push' it back in when it is needed again. The answer to this depends upon the type of Internet connection being used.

In T1, DSL, DDS, etc. connections, the interface will usually change to a DOWN state when the Internet connection is down. When the interface changes state to DOWN, all routes using that interface are then automatically removed from the route table. When the interface returns to an UP state, the routes are re-inserted back in.

In Ethernet hand-off connections, and in some failure states of the previous connection types, the interface will stay UP, even though IP connectivity is lost. Without the interface going DOWN, the router has no way to automatically remove the routes directing traffic out of those interfaces. Network Monitor was created to address this issue, and is covered in the next section.

Interface Tracking (if applicable)

If the Internet interface will not go down, or detection of lost IP connectivity is essential, interface tracking is required. This process is called 'Network Monitor'. Network Monitor allows the router to 'probe' another device through the use of a ping, TCP connection, or HTTP RAW. In this case, ping probes will be used. The probe is then monitored by a 'track' that can be applied to various functions of the configuration to dynamically remove parts of it from the router's use when the track is in a failed state. In this case, it will be applied to a route.

There are several steps to configuring Network Monitor, listed here:

- Configure the probe(s)
- Configure the tracks(s)
- Force the traffic out the correct interface
- Apply the tracks to the configuration

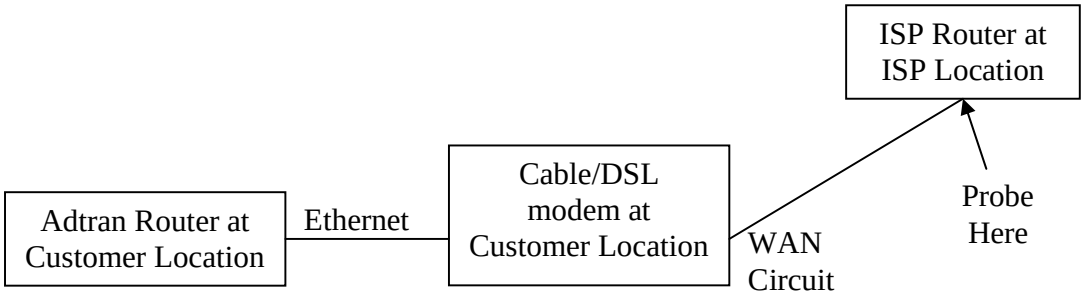
Configuring the Probe

What should be probed? The answer depends on the situation. The best practice is to ping the ISP's side of the Internet connection. This is not always possible, as in the case of a dynamic connection, or preferable, as in the case of a particular server that the router or clients behind it have to be able to connect to.

The configuration of a probe involves specifying the destination IP address or hostname, specifying the source IP address (optional), specifying the period between probes, and specifying the number of times a single test must fail before the probe fails. The time to detect a change then follows this equation:

$$Link\_Detection\_Time(\text{sec}) = Period \times Consecutive\_Interval$$

In the most common situation where Network Monitor is required, a modem is terminating a cable or DSL WAN circuit and is handing off Ethernet to the customer. For this situation it is advised to probe the ISP's router at the ISP's location. This is due to the fact that most cable and DSL modems will not bring down the Ethernet interface of the modem when the WAN circuit goes down so the next logical upstream device to ping would be the ISP router on the other end of the WAN circuit. The assumption is made here that if a ping can get to the ISP that it can get beyond the ISP. A diagram is shown here:



A sample configuration of a probe is shown here:

```
probe WAN1 icmp-echo
  destination <IP Address or Hostname>
  source <IP Address (Optional)>
  period <interval between tests (seconds)>
  tolerance consecutive fail <# needed to fail> pass <# needed to pass>
  no shutdown
```

Configuring the Track

A track's primary function is to monitor various tests, weigh them, and determine a pass or failed state. In this case, only probes will be tracked, but this function has the ability to track other variables, such as schedules, as well. A track also has the ability to apply a dampening interval that is designed to protect against flapping in certain situations. In this case, this property will not be utilized because flapping has already been accounted for by the consecutive failure property of the probe.

A sample configuration of a track is shown here:

```
track WAN1
  test if probe WAN1
  no shutdown
```

Force the Traffic Out the Correct Interface

Since this entire process is designed to dynamically change the routing table, a mechanism must be in place to force probes out the correct interface when the default route is not out that interface. The two methods of accomplishing this task are through static routing or Policy-Based Routing (PBR).

When static routing is utilized, a host route (/32 mask) is created so that the route table will always send the traffic out the specified interface, and the route table agrees with the probe traffic. In the event the interface goes down, a backup route to interface 'null 0' is created so that the probe is not incorrectly created out another interface.

An example static routing configuration for a statically assigned Ethernet interface is shown here, complete with a probe & track for reference:

```
interface eth 0/1
  ip address <WAN1 IP Address> <WAN1 Subnet Mask>
!
probe WAN1 icmp-echo
  destination <IP Address of the ISP's router at ISP's location>
  source <WAN1 IP Address>
  period <interval between tests (seconds)>
  tolerance consecutive fail <# needed to fail> pass <# needed to pass>
  no shutdown
!
track WAN1
  test if probe WAN1
  no shutdown
!
ip route <Probe Destination> 255.255.255.255 <WAN1 Gateway IP>
ip route <Probe Destination> 255.255.255.255 null 0 10
```

PBR is a feature-rich command set that allows the administrator to force the router to make routing decisions independent of the current routing table. It is sometimes referred to as "Source Based Routing", because it allows the router to match upon an access-list specifying source addresses of interest. An extended access-list can specify not only the source address, but destination address, source & destination ports, and the protocol type.

In this case, PBR will be applied to the 'local' policy, which governs all traffic generated from the router itself. An access-list will match the traffic that is generated by the probe, and a route-map will force the traffic out a particular interface or next-hop IP address.
In the case of a statically set Ethernet interface it has no clearly defined peer, so next-hop IP address should be used here in place of interface. In the case of a dynamically addressed Ethernet interface, the interface remembers its learned default gateway, and treats that as the peer when it is set to interface. This is a little confusing, but it makes sense that a dynamically assigned Ethernet interface could not use next-hop IP address because the next-hop could not be predicted. In the case of statically set Ethernet interface, the gateway is known to the administrator, but not the router, so next-hop IP

address must be used. As with the static routing example, a backup to ‘null 0’ should be used to avoid an invalid session being opened by the firewall.

Since the route table will not agree with the probe traffic, the Reverse Path Forwarding (RPF) check must be disabled to avoid the possibility that the return probe traffic is not dropped while the backup connection is active. The RPF check is commonly referred to as the ‘Spoofing’ check.

A sample PBR configuration for a statically assigned Ethernet interface is shown here, complete with a probe & track for reference:

```
interface eth 0/1
  ip address <WAN1 IP Address> <WAN1 Subnet Mask>
!
probe WAN1 icmp-echo
  destination <IP Address of the ISP’s router at ISP’s location>
  source <WAN1 IP Address>
  period <interval between tests (seconds)>
  tolerance consecutive fail <# needed to fail> pass <# needed to pass>
  no shutdown
!
track WAN1
  test if probe WAN1
  no shutdown
!
ip access-list extended WAN1
  permit icmp host <WAN1 IP Address> host <IP Address of the ISP’s
      router at ISP’s location>
!
route-map LOCAL permit 10
  match ip address WAN1
  set ip next-hop <WAN1 Gateway IP>
  set interface null 0
!
ip local policy route-map LOCAL
!
no ip policy-class Public rpf-check
```

The next sample config will be of a dynamically set Ethernet interface. This configuration will be slightly different because neither the IP address of the interface, nor the IP address of the gateway can be known in advance. Because of this, the gateway cannot be probed. An Internet address that is trusted not to fail, like www.google.com, will be probed. If www.google.com would ever to go down, this Internet connection would go down as well even if all other websites were working. To get around this problem, multiple websites will be probed and compared. If the AOS router will be pinging hostnames instead of manually configured IP addresses a DNS server must be specified in the configuration with the **ip name-server** command. All must fail for the track to fail. It is shown here:

```
ip name-server 208.61.209.1 208.61.209.2
!
interface eth 0/1
  ip address dhcp track WAN1
!
probe WAN1-Test1 icmp-echo
  destination www.google.com
  period <interval between tests (seconds)>
  tolerance consecutive fail <# needed to fail> pass <# needed to pass>
  no shutdown
!
probe WAN1-Test2 icmp-echo
  destination www.yahoo.com
  period <interval between tests (seconds)>
  tolerance consecutive fail <# needed to fail> pass <# needed to pass>
  no shutdown
!
probe WAN1-Test3 icmp-echo
  destination www.ask.com
  period <interval between tests (seconds)>
  tolerance consecutive fail <# needed to fail> pass <# needed to pass>
  no shutdown
!
track WAN1
  test list or
    if probe WAN1-Test1
    if probe WAN1-Test2
    if probe WAN1-Test3
  no shutdown
!
ip access-list extended WAN1
  permit icmp any hostname www.google.com
  permit icmp any hostname www.yahoo.com
  permit icmp any hostname www.ask.com
!
route-map LOCAL permit 10
  match ip address WAN1
  set interface eth 0/1 null 0
!
ip local policy route-map LOCAL
!
no ip policy-class Public rpf-check
```

Apply the Track to the Configuration

Up to this point, the configuration of the router has not been affected. By applying the track to a route, it will dynamically remove or insert the route depending upon the current state of the track. The syntax for a route statement with a track is:

ip route <*Destination IP Address*> <*Destination Mask*> <*Next-Hop IP Address or Interface*> **track** <*Track Name*> <*Administrative Distance*>

A sample configuration is shown here:

```
ip route 0.0.0.0 0.0.0.0 <WAN1 Peer IP Address OR WAN1 Interface> track WAN1
!
ip route 0.0.0.0 0.0.0.0 <WAN2 Peer IP Address OR WAN2 Interface> 10
```

Dynamically assigned interfaces which learn the default route dynamically must apply the track along with the ip address statement on the interface, as shown here:

PPP Interface:

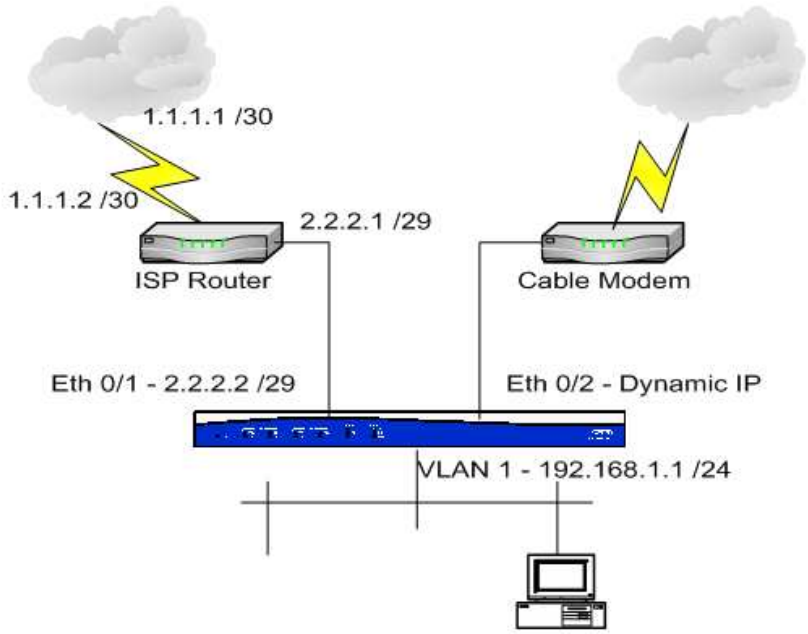
ip address negotiated <*administrative distance*> **track** <*track name*>

Ethernet Interface:

ip address dhcp <*administrative distance*> **track** <*track name*>

Configuration Example

In this example, there will be two internet connections. The primary is a statically addressed internet connection through an ISP's router. The backup is a dynamically addressed connection through a cable modem. A network diagram is shown here:



The configuration for this router is shown here:

```
ip local policy route-map LOCAL
!
probe WAN1 icmp-echo
  destination 1.1.1.1
  source 2.2.2.2
  period 3
  tolerance consecutive fail 3 pass 3
  no shutdown
!
track WAN1
  test if probe WAN1
  no shutdown
!
ip access-list extended WAN1
  permit icmp host 2.2.2.2 host 1.1.1.1
!
route-map LOCAL permit 10
  match ip address WAN1
  set ip next-hop 2.2.2.1
  set interface null 0
!
interface eth 0/1
  ip address 2.2.2.2 /29
  access-policy Public
  no shutdown
!
interface eth 0/2
  ip address dhcp 10
  access-policy Public-Backup
  no shutdown
!
interface vlan 1
  ip address 192.168.1.1 /24
  access-policy Private
  no shutdown
!
ip access-list extended self
  remark Traffic to Netvanta
  permit ip any any
!
ip access-list extended NAT
  remark NAT all to the primary connection
  permit ip any any
!
ip access-list extended NAT-Backup
  remark NAT all to the secondary connection
  permit ip any any
!
ip access-list extended AdminAccess
  remark Outside Admin Access
  permit tcp any any eq ssh
  permit tcp any any eq https
!
```

```
ip policy-class Private
  allow list self self
  nat source list NAT interface eth 0/1 policy Public
  nat source list NAT-Backup interface eth 0/2 policy Public-Backup
!
no ip policy-class Public rpf-check
ip policy-class Public
  allow list AdminAccess self
!
ip policy-class Public-Backup
  allow list AdminAccess self
!
ip route 0.0.0.0 0.0.0.0 2.2.2.1 track WAN1
```