Technical Guide

Tips and Tricks | AlliedWare Plus™ Managed Switches

Introduction

This document contains useful technical tips and tricks for AlliedWare Plus Managed switches.

These Tips and Tricks apply to:

- SwitchBlade™ x8100 Series
- SwitchBlade x908
- x900 Series switches
- x610 Series switches
- x600 Series switches
- x510 Series switches
- x310 Series switches
- x210 Series switches

New in this Revision

AlliedWare Plus™ OPERATING SYSTEM

Contents

Management

Using shell scripts

AlliedWare Plus supports shell scripts. You can use this powerful interface for information gathering and device configuration.

Note: Shell scripts must have the file extension `.sh`.

This section describes a script that configures an IP interface, sets switch ports to trunk mode, executes show commands and returns output to the terminal.

Note: This script does not contain statically configured interface names and IP addresses. Instead, you enter these as command arguments when the script is executed. This allows you to re-use the script. You could develop a collection of scripts that allow you to perform frequent tasks quickly and efficiently.

When you run this script, you must enter three parameters at the command line:

1. the VLAN ID to be created
2. the IP address to be assigned to the VLAN
3. the switch ports to be added to the VLAN

The script

The script is named `vlan-port-ip.sh` and contains:

```
# configure VLAN, add an IP
echo "Configuring VLAN and IP"
echo -e "
 enable\n
 configure terminal\n
 vlan database\n
 vlan $1\n
 exit\n
 interface vlan$1\n
 ip address $2\n
" | imish

# Assign switch ports to VLAN
echo "Configuring Switch Ports"
echo -e "
 enable\n
 configure terminal\n
 interface $3\n
 switchport access vlan $1
```

```
" | imish

# show ip interfaces
echo -e "
 show ip int brief\n
 " | imish
```

Running the script

This example uses the script to create vlan120, assign it an IP address of 192.168.1.120/24, and put ports 1.0.10 and 1.0.11 into it. Enter Privileged Exec mode and use the command:

```
awplus#activate vlan-port-ip.sh 120 192.168.1.120/24
   port1.0.10-port1.0.11
```

The script returns the following output to the console:

```
Configuring VLAN and IP

AlliedWare Plus (TM) 5.2.1 07/27/07 00:44:25

Enter configuration commands, one per line.  End with CNTL/Z.



Configuring Switch Ports

AlliedWare Plus (TM) 5.2.1 07/27/07 00:44:25

Enter configuration commands, one per line.  End with CNTL/Z.

AlliedWare Plus (TM) 5.2.1 07/27/07 00:44:25

Interface             IP-Address      Status          Protocol
eth0                  172.28.8.220    admin up        running
vlan120               192.168.1.120   admin up        down
```

Verifying the configuration

You can verify the configuration by checking the running-config. The following shows the relevant parts of the resulting running-config:

```
awplus# show run

vlan database
 vlan 120 state enable
!
interface port1.0.10-1.0.11
 switchport mode access
 switchport access vlan 120
!
interface vlan120
 ip address 192.168.1.120/24
!
```

Using command scripts

Command scripts are supported in AlliedWare Plus.

Command scripts are different to device configuration files.

Note: **Command scripts must not have the file extension .sh. We recommend using .scp.**

This section describes a script that creates a VLAN with ID number 2, names it "video2", and assigns the IP address 192.168.2.1 with a class C mask. The script contains the same commands as you would enter at the command line.

The script

The script is named `vlan2.scp` and contains:

```
enable
conf t

vlan database
vlan 2 name video2

interface vlan2
ip address 192.168.2.1/24

end
```

Note: You must include the commands `enable`, `conf t`, and `end` in the script.

Running the script

To run the script, enter Privileged Exec mode and use the command:

```
awplus#activate vlan2.scp
```

The script returns the following output to the console:

```
AlliedWare Plus (TM) 5.2.1 07/20/07 00:45:15

Enter configuration commands, one per line.  End with CNTL/Z.

awplus#
```

Verifying the configuration

You can verify the configuration by checking the running-config. The following figure shows the relevant parts of the resulting running-config:

```
awplus# show run

vlan database
 vlan 2 name video2
 vlan 2 state enable
!
interface vlan2
 ip address 192.168.2.1/24
```

Large configurations in the "conf t" environment

The issue

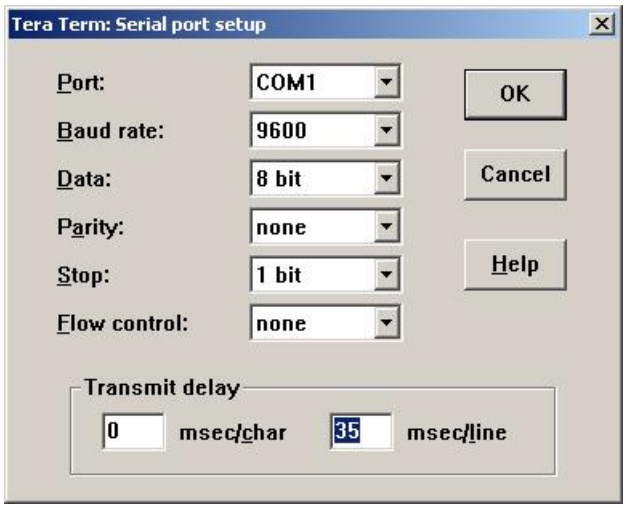
Pasting in very large configurations using the console at the "conf t" prompt can give unpredictable results. Consoles, as standard practice, do not have flow control. If too much text is pasted, it will exhaust the buffer size available for the console.

The solution

There are three possible options:

1. **The best practice is to copy in as a file using TFTP.**
2. If you do have to paste to conf t, you can void the issue by breaking the configuration down into smaller portions, and pasting in a portion at a time.
3. Another practical solution is to change your terminal program's setting to introduce an end line delay period. One example, using Linux minicom, involves setting a Newline Delay (using Ctrl-a, t, d) of at least 150ms to fix the issue.

Also, hyperterminal offers this setting on connection:



Using SFTP to transfer files to/from an AlliedWare Plus switch

Introduction

Secure File Transfer Protocol (SFTP) is a file copy protocol that is supported by the Secure Shell (SSH) service in AlliedWare Plus. By default, when SSH is enabled on a switch running AlliedWare Plus, SFTP is also enabled.

You can see whether the service is enabled by using the show ssh server command:

```
awplus#show ssh server
Secure Shell Server Configuration
-----------------------------------------------------------
SSH Server                          : Enabled
Protocol                            : IPv4,IPv6
Port                                : 22
Version                             : 2,1
Services                            : scp, sftp  <-------------
User Authentication                 : publickey, password
Resolve Hosts                       : Disabled
Session Timeout                     : 0 (Off)
Login Timeout                       : 60 seconds
Maximum Startups                    : 10
Debug                               : NONE
```

You can enable or disable the service using the command:

```
(no) ssh server sftp
```

The popular FTP client Filezilla can operate as an SFTP client. This provides a convenient graphical interface for transferring files to or from a switch running AlliedWare Plus.

Configuring the switch

There are three steps to enabling SSH server on the switch:

1. Create a hostkey:

```
awplus(config)#crypto key generate hostkey rsa
Generating host key (1024 bits rsa)
This may take a while. Please wait ... Done
WARNING: The SSH server must now be enabled with "service ssh"
```

2. Enable SSH Server:

```
awplus(config)#service ssh
WARNING: SSHv1 host key does not exist. SSH will not be available for
version 1.
```

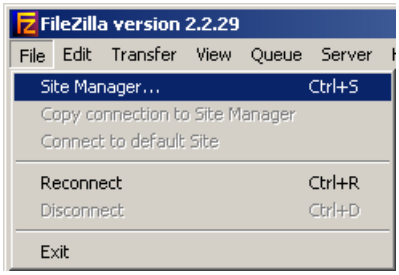
3. Enable one or more users to access SSH:

```
awplus(config)#ssh server allow-users manager
```

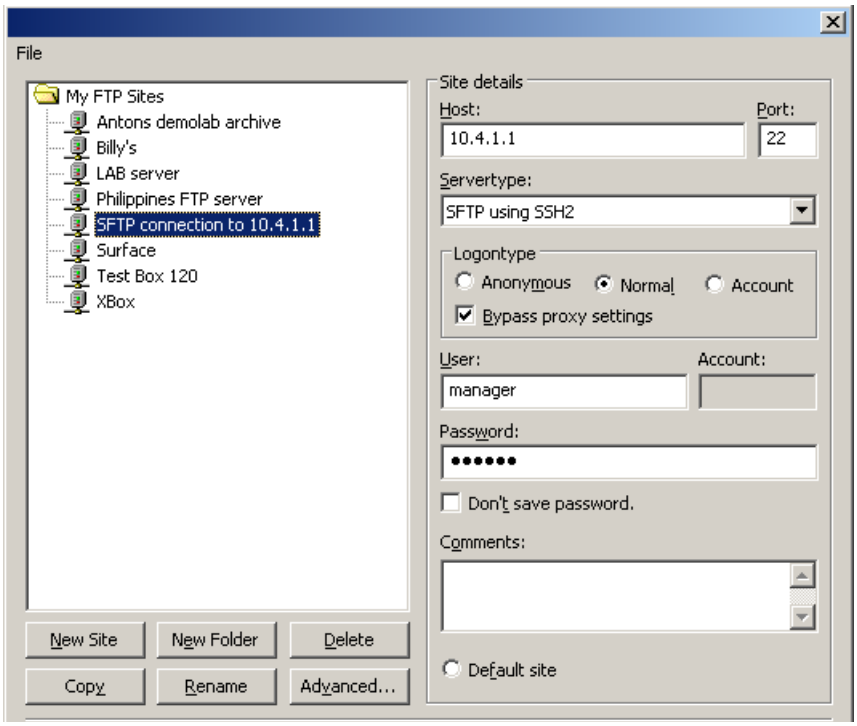
Configuring Filezilla

Within Filezilla, you need to create an FTP site definition that uses SFTP to connect to your switch.

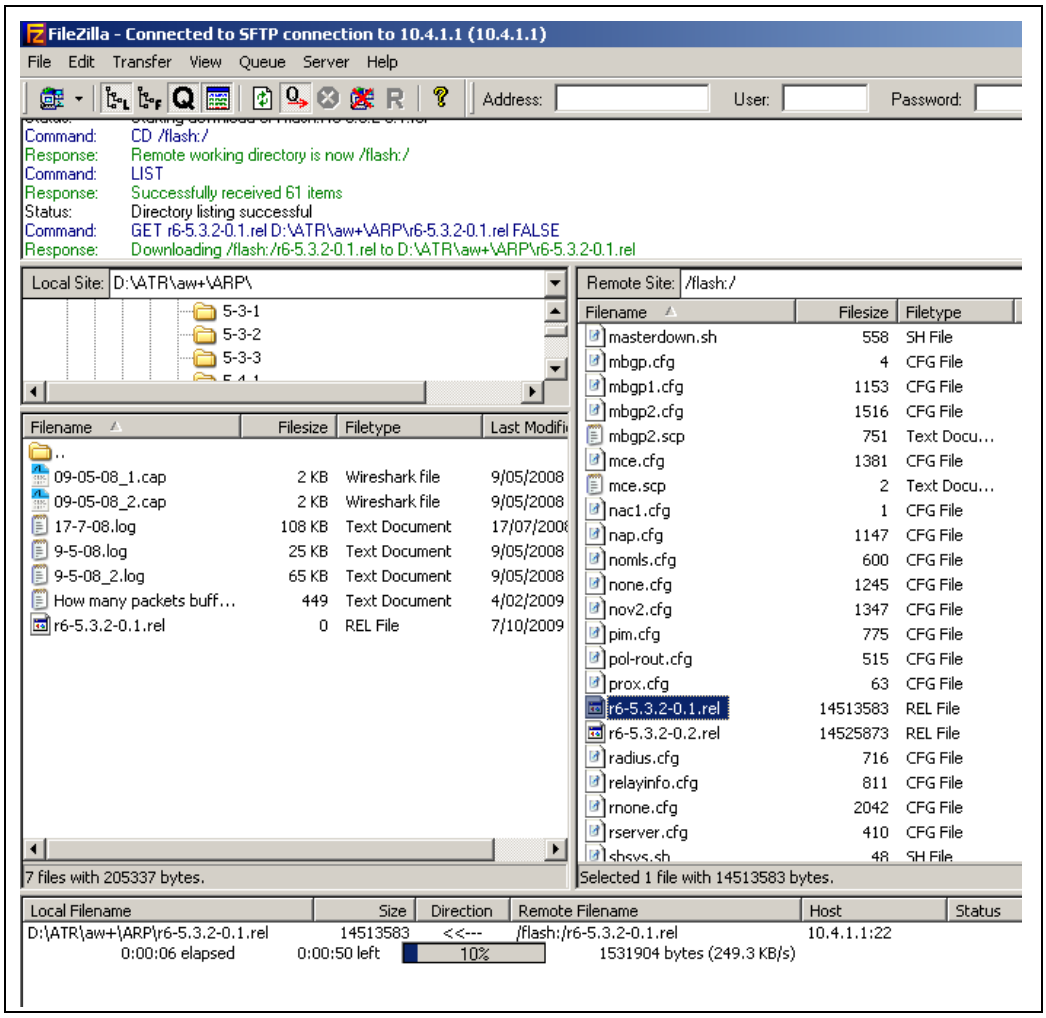
1. Select **File > Site Manager….**



2. Create an FTP site that uses the Servertype **SFTP using SSH2**. Filezilla automatically selects port 22 as the TCP port for this FTP site:



3. Connect to the site. The contents of the file system on the switch are displayed in the Remote Site pane. Files can be transferred to and from the switch in the same way as they can be transferred to any FTP site by Filezilla:



Using RSA to securely copy files to and from the switch

To securely copy files to and from a switch, use RSA private/public key authentication.

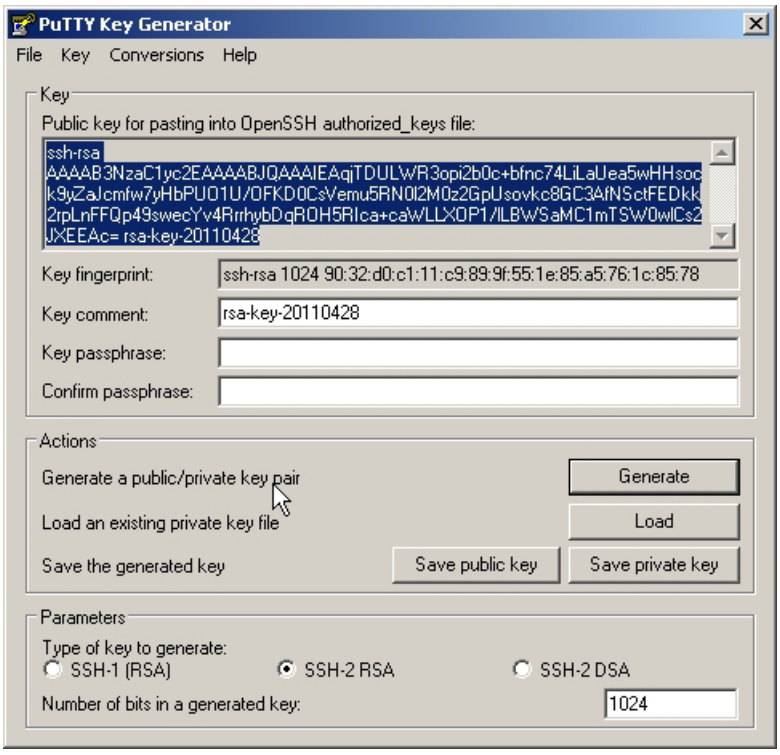
This example uses the Putty suite of secure device management and file transfer tools. You can download these tools, puttygen.exe, psftp.exe and pscp.exe, at:

http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

On the Windows client

First, generate the RSA Private/Public key pair. This is done using PuTTY Key Generator:

1. In PuTTY, use the **Generate** button to generate the keys.
2. Next, use the **Save public key** and **Save private key** buttons to save the public and private keys to separate files, for example user1.pub and user1.ppk.



Copy the public key user1.pub onto an SD card so that it can be transferred to the switch. You can also use TFTP to transfer this file to the switch.

On the switch

1. Create the two private RSA keys which are required for each type of SSH version:

```
awplus(config)# crypto key generate hostkey rsa
awplus(config)# crypto key generate hostkey rsa1
```

2. Enable the SSH server:

```
awplus(config)#service ssh
```

3. Create the SSH user:

```
awplus(config)#username steve privilege 15 password secret
```

4. Register the user as an SSH client:

```
awplus(config)#ssh server allow-users user1
```

5. Copy the client's public key onto the switch from the SD card (or use TFTP):

```
awplus(config)#copy card:user1.pub flash:
```

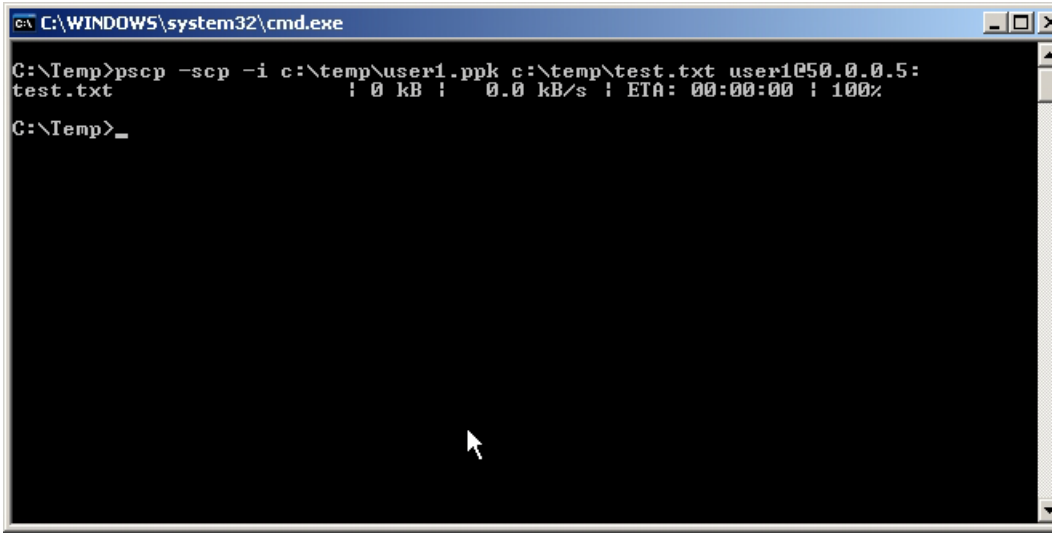
6. Associate the public key file with the SSH user:

```
awplus(config)#crypto key pubkey-chain userkey user1 user1.pub
```

The switch is now ready to accept SCP and SFTP connections from user User1.

Using SCP to securely copy files - on the client

1. Open a command prompt box.
2. Navigate to the folder which contains the pscp.exe program and the public and private key files.
3. Use pscp.exe to login and copy a file onto the switch, as shown below:



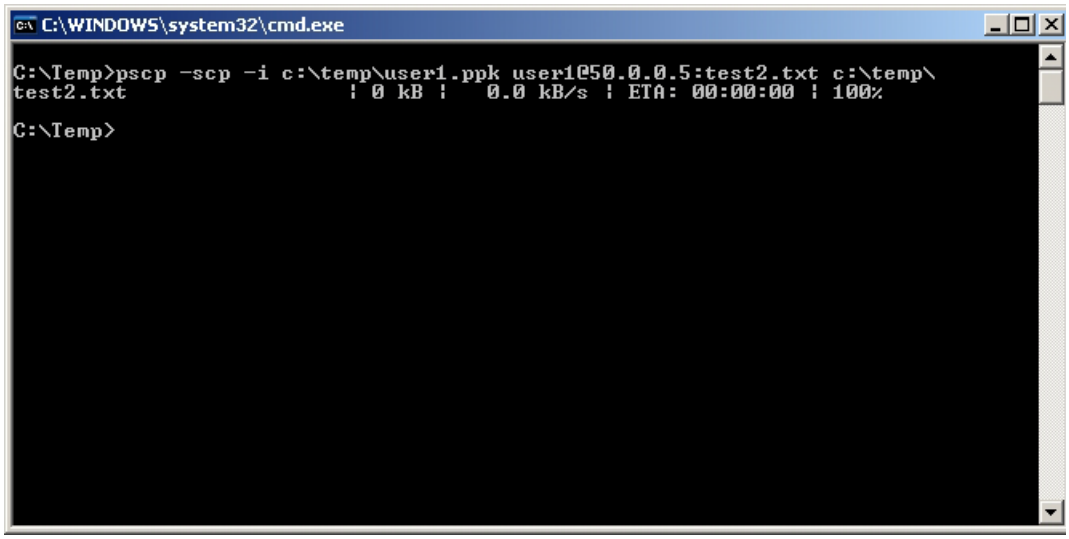
The command:

```
pscp -scp -i c:\temp\user1.ppk c:\temp\test.txt user1@50.0.0.5:
```

includes the following parameters:

PARAMETER	DESCRIPTION
-scp	tells the program to use SCP instead of SFTP
-i c:\temp\user1.ppk	the location of the private key file
c:\temp\test.txt	the location of the file to be copied to the switch
user1@50.0.0.5:	the SSH username, at the IP address of the switch

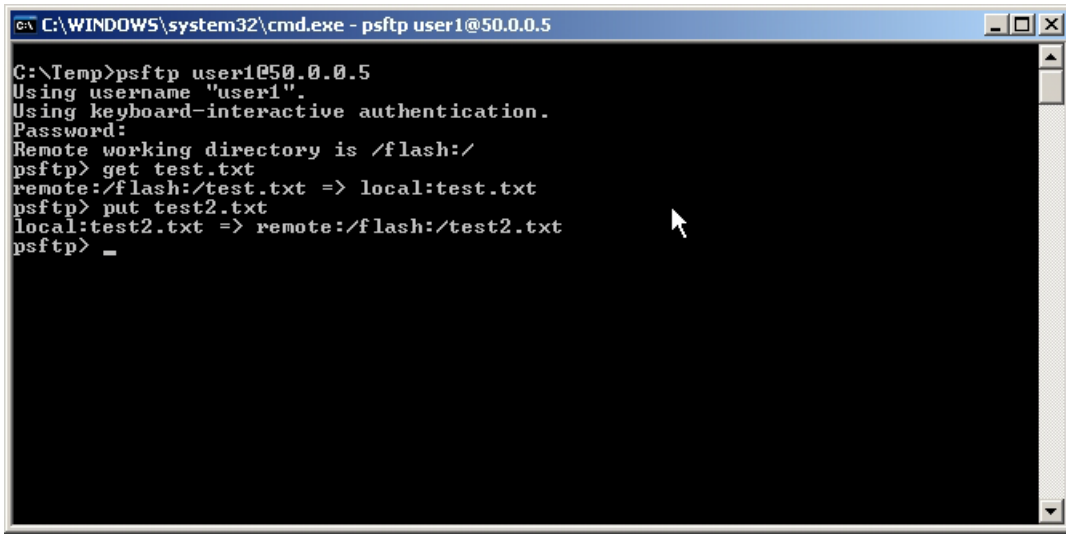
Use pscp.exe to login and copy a file from the switch to the client:




```
pscp -scp -i c:\temp\user1.ppk user1@50.0.0.5:test2.txt
   c:\temp\
```

Using SFTP to securely copy files - on the client

1. Login to the switch using psftp.exe



The syntax is:

```
psftp <SSH username>@<IP address of the switch>
```

You are now prompted for the password associated with the SSH user:

- To copy from the client to the switch, use the **put** command.
- To copy from the switch to the client, use the **get** command.

Switching

How to view switch tables

You can view the contents of switch tables with the command:

```
show platform table <table-name>
```

Commonly used tables are:

THIS KEYWORD...	DISPLAYS...
fdb	the platform forwarding database table
ip	the platform IP table
ipmulti	the platform IP multicast table
l2mc	the platform L2 multicast table
macfull	the full platform MAC table—all MAC addresses that the switch has learned
port counters	counters from the platform port table—see the following section

How to view port counters

You can view port counters with the command:

```
awplus#show platform table port counters
```

```
Switch Port Counters
----------------------------------------------------------------------

Port 1.0.1    Ethernet MAC counters:
 Combined receive/transmit packets by size (octets) counters:
  64                              0 512 - 1023                          0
  65 - 127                        0 1024 - MaxPktSz                     0
  128 - 255                          0
  256 - 511                          0
 General Counters:
 Receive                             Transmit
  Octets                          0 Octets                              0
  Pkts                            0 Pkts                                0
  CRCErrors                          0
  MulticastPkts                   0 MulticastPkts                       0
  BroadcastPkts                   0 BroadcastPkts                       0
  FlowCtrlFrms                    0 FlowCtrlFrms                        0
OversizePkts                         0
  Fragments                          0
  Jabbers                            0
  UpsupportOpcode                    0
  UndersizePkts                      0
Collisions                           0
LateCollisions                       0
  ExcessivCollsns                    0
 Miscellaneous Counters:
  MAC TxErr                          0
  MAC RxErr                          0
  Drop Events                        0
```

Adding a VLAN to an LACP trunk causes port flapping

The situation

LACP trunks can be configured to allow all VLANs, as shown in the below example. In this situation, when you add another VLAN it causes port flapping.

```
!
interface port1.0.2
 switchport
 switchport mode trunk
 switchport trunk allowed vlan all
 switchport trunk native vlan none
 channel-group 1 mode active
 lacp timeout long
!
interface port1.0.4
 switchport
 switchport mode trunk
 switchport trunk allowed vlan all
 switchport trunk native vlan none
 channel-group 1 mode active
 lacp timeout long
!
```

The reason

When you add a new VLAN to the LACP trunk, the VLAN is automatically added to each of the ports in the trunk. This causes the ports to mismatch, and so as they are configured they are first removed and then re-added to the LACP aggregator. When this happens, the ports briefly go down and then come back up, which causes a very short interruption to traffic.

MTU/MRU commands

ITEM	DESCRIPTION
MRU	Maximum Receive Unit. This is the maximum L2 frame size an ingress port may receive
MTU	Maximum Transmission Unit. This signifies the maximum L3 packet size a given L3 interface can transmit
Jumboframe	The jumboframe setting allows ports on a given switch to receive jumbo frames.

Observed behaviours

1. On AlliedWare Plus switches, the MTU setting on a L3 interface is adhered to by the hardware as well. When the hardware detects a transmitted packet that is too big for the L3 MTU, the packet is sent to the CPU.
2. On AlliedWare Plus switches, if the IP stack attempts to forward a packet that is too big for the VLAN MTU, it sends an ICMP Too Big message to the sender.
3. On AlliedWare Plus switches, you can only set the MTU on L3 interfaces (VLANs).
4. On AlliedWare Plus switches, the MTU command is:
   syntax: mtu <68-1500>
   syntax: no mtu
   default: mtu are set to 1500
   mode: interface mode (ports only)

Note: The maximum MTU is 1500 because although the silicon is capable of switching L3 packets bigger than 1500, we do not currently support software forwarding of packets larger than 1500.Support for this is planned for the future.

5. On x600, x610, x510, and x210 switches, MTU is implemented as part of a port characteristics, so setting an MTU value for a VLAN sets the MTU for all the VLAN's member ports. As such, you can only set the MTU for VLAN's whose members are non-trunked (do not belong to any other VLANs).
6. On x600, x610, x510, and x210 switches, there is an MRU command:

```
syntax: mru <68-16375
syntax: no mru
default: mru are set to 16383 (16375 + 8)
mode:   interface mode (ports only)
```

Note: The maximum MRU is 16375 which is 16383 - 4 bytes for VLAN tag and - 4 bytes for CRC.

7. On x900 and x980 switches, enabling the Jumboframe setting sets the MRU of all ports to 10240 bytes. However enabling the Jumboframe setting does not automatically set the MTU
8. On x900 and x908 switches, you cannot set the MRU for individual ports.
9. On AlliedWare Plus switches, the MRU setting is only shown for ports, and MTU settings are only shown for VLANs.
10. On x600, x610, x510, and x210 switches, the default:
    - user MRU is 1500
    - hardware L2 MRU is 1522 (1500 + 22 for eth headers)
    - hardware L2 MTU is 1526 (1500 + 22 for eth headers + 4 byte tag)
    - user MTU is 1500
    - hardware L3 MTU is 1504 (1500 + 4 byte tag)
11. On x900 and x908 switches, the default:
    - hardware L2 MRU is 1522 when jumboframe mode is off (10240 when jumboframe mode is on)

- user MTU is 1500
- hardware L3 MTU is 1500

Note: On x900 and x908 series, L3 MTU setting is part of route structures. When the L3 MTU setting changes, hardware routes are deleted and repopulated with routes with the new MTU.

```
Interface port1.0.1
  Scope: both
  Link is DOWN, administrative state is UP
  Thrash-limiting
    Status Not Detected, Action learn-disable, Timeout 1(s)
  Hardware is Ethernet, address is 0015.77c9.73a1
  index 5001 metric 1 mru 1522
  <UP,BROADCAST,MULTICAST>
  VRF Binding: Not bound
  SNMP link-status traps: Disabled
    input packets 0, bytes 0, dropped 0, multicast packets 0
    output packets 0, bytes 0, multicast packets 0 broadcast packets 0

awplus#sh int vlan1
Interface vlan1
  Scope: both
  Link is UP, administrative state is UP
  Hardware is VLAN, address is 0015.77c9.73a1
  IPv4 address 172.20.5.109/15 broadcast 172.21.255.255
  index 201 metric 1 mtu 1500
  arp ageing timeout 300
  <UP,BROADCAST,RUNNING,MULTICAST>
  VRF Binding: Not bound
  SNMP link-status traps: Disabled
  Bandwidth 1g
    input packets 115, bytes 10090, dropped 0, multicast packets 0
   output packets 98, bytes 4550, multicast packets 0 broadcast packets
0
```

Using QoS to mark packets' DSCP value, and assign them to a queue

This applies to x600, x610, x510, and x210 Series switches.

In this example you want to achieve the following:

- Mark pings from 10.0.0.1 to 10.0.0.2 with DSCP 46, and assign them to egress queue 5
- Mark pings from 10.0.0.1 to 10.0.0.13 with DSCP 34, and assign them to egress queue 3
- Mark telnet from 10.0.0.1 to 10.0.0.13 with DSCP 26, and assign them to egress queue 2
- Mark all other traffic with DSCP 18, and assign them to egress queue 2

To do this, use the following configuration.

1. First create the appropriate access lists that will match on the various types of traffic and their source and destination:

   An access list to match on pings from 10.0.0.1 to 10.0.0.2:

   ```
   access-list hardware ping1
   permit icmp 10.0.0.1/32 10.0.0.2/32 icmp-type 8
   ```

   An access list to match on pings from 10.0.0.1 to 10.0.0.13

   ```
   access-list hardware ping2
   permit icmp 10.0.0.1/32 10.0.0.13/32 icmp-type 8
   ```

   An access list to match on telnet from 10.0.0.1 to 10.0.0.13

   ```
   access-list hardware telnet1
   permit tcp 10.0.0.1/32 10.0.0.13/32 eq 23
   ```

2. Then create class-maps that match on the access-lists:

   ```
   class-map ping1
   match access-group ping1

   class-map ping2
   match access-group ping2

   class-map telnet1
   match access-group telnet1
   ```

3. Next, create a policy-map and configure class-maps under it to remark the DSCP values and assign egress queues.

On the x600, x610, x510, and x210 switches, DSCP values cannot be premarked in packets prior to policing. They can only be remarked after policing.

The command **police single-rate <cir> <cbs> <ebs> action remark-transmit** must be used within the actions for each of the class-maps within the policy-map in this example, in order for the DSCP value to be remarked.

Given that you do not want to actually rate limit the traffic at all, use the maximum value for each of the following:

- CIR (Committed Information Rate)- 1-16000000 kbps
- CBS (Committed Burst Size) - (0-16777216 bytes
- EBS (Excess Burst Size) - 0-16777216 bytes

Then, to perform the actual remarking of the DSCP values, use the command **remark-map to new-dscp** x

This example also uses the **remark new-cos internal** command.

This command will effectively assign packets to egress queues. The "internal" CoS value is not actually written into the packets, it is just used as a lookup in the cos-to-queue map, to choose the packet's egress queue.

By default, CoS values are mapped to queues as follows:

CoS value	0	1	2	3	4	5	6	7
Egress Queue No	2	0	1	3	4	5	6	7

For example, the command **remark new-cos 2 internal** assigns the packet to Egress Queue 1.

```
policy-map qos-test
class default
  remark new-cos 0 internal
   remark-map to new-dscp 18
   police single-rate 16000000 16777216 16777216 action remark-transmit

class ping1
   remark new-cos 5 internal
   remark-map to new-dscp 46
   police single-rate 16000000 16777216 16777216 action remark-transmit

class ping2
   remark new-cos 4 internal
   remark-map to new-dscp 34
   police single-rate 16000000 16777216 16777216 action remark-transmit

class telnet1
   remark new-cos 3 internal
   remark-map to new-dscp 26
   police single-rate 16000000 16777216 16777216 action remark-transmit
```

4. Finally add the policy-map to the port with the **service-policy** command:

```
interface port1.0.2
switchport
switchport mode access
service-policy input qos-test
```

Unable to use the multicast address 232.x.x.x without specifying a source

The issue

If the Source Specific Multicast (SSM) multicast address 232.x.x.x is used for a stream, and the client does not send a source address in the request for this group, the switch discards this request. It does not create a entry on the L2MC table when it receives these IGMP reports.

This is correct behaviour

Group addresses 232.0.0.0/8 are reserved for the SSM range.

SSM is a method of multicasting where the client (receiver) requests a multicast group from a specific source only. This reduces the amount of multicast routing information required, as the network does not have to discover multiple multicast sources.

Therefore, if the client does not specify a source address for a group in the 232.0.0.0/8 range, the switch will not register anything as a result of receiving this packet.

IGMPv2 backward compatibility

However, the fact is that there is a large installed base of equipment that supports on IGMPv2, and not IGMPv3. It would be extremely annoying to not be able to use SSM in a network simply because some of the equipment connected to it does not support IGMPv3.

Consider the case of a service providing delivering TV as multicast over Ethernet. If this provider is receiving content from upstream content providers who only support PIM SSM and will not accept any (*,G) joins, then the service provider must implement PIM SSM in their network. However, it is highly likely that at least some of the subscribers connected to the network will be using Set Top Boxes that are not capable of IGMPv3. This service provider is then stuck between a rock and a hard place, they either need to go around and replace ALL subscribers' Set Top Boxes with IGMPv3 capable devices, or they need their content providers to relax their (S,G) join requirements. Neither of these options is going to be easy.

Fortunately, there is a third option, the multicast routers in the network could help them out, and provide a work-around that converts IGMPv2 reports into Source-Specific reports.

This third option is exactly what AW+ provides.

To configure this feature, proceed as follows:

1. Create an access-list to define a range of multicast group addresses.

```
access-list 10 permit 232.1.67.0 0.0.0.255
```

2. Enable SSM mapping of IGMPv1/v2 reports.

```
ip igmp ssm-map enable
```

IPv6 on AlliedWare Plus - operation with a PC

You can configure a switch to operate with a PC that has IPv6 enabled. The steps are as follows:

1. Enable IPv6 forwarding:

```
awplus(config)#ipv6 forwarding
```

2. Configure an IPv6 address on the VLAN in which the PC is connected (VLAN1 in this case):

```
awplus(config-if)# ipv6 address 2001:1111::1/64
```

3. Enable Router Advertisement (RA) for IPv6 stateless configuration on the interface (these are disabled by default):

```
awplus(config-if)# no ipv6 nd suppress-ra
```

4. Specify the IPv6 prefix that is advertised for IPv6 address auto-configuration:

```
awplus(config-if)# ipv6 nd prefix 2001:1111::/64
```

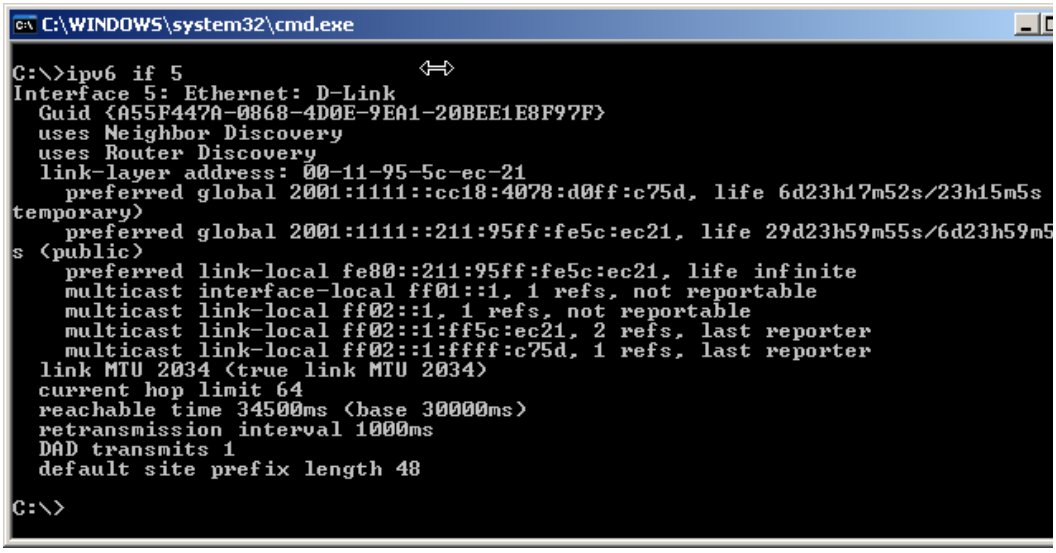
Use the **show ipv6 neighbors** command to see the PC connected. The below example shows port1.0.1 in VLAN. It also shows the PC's Preferred Global (temporary) IPV6 address:

```
awplus#show ipv6 neighbors
IPv6 Address                          MAC Address    Interface Port      Type
  sta = static   dyn = dynamic
2001:1111::cc18:4078:d0ff:c75d          0011.955c.ec21 vlan1
port1.0.1  dyn
```

The following shows the PC's Preferred Link-local IPv6 address:

```
fe80::211:95ff:fe5c:ec21            0011.955c.ec21 vlan1    port1.0.1
dyn
```

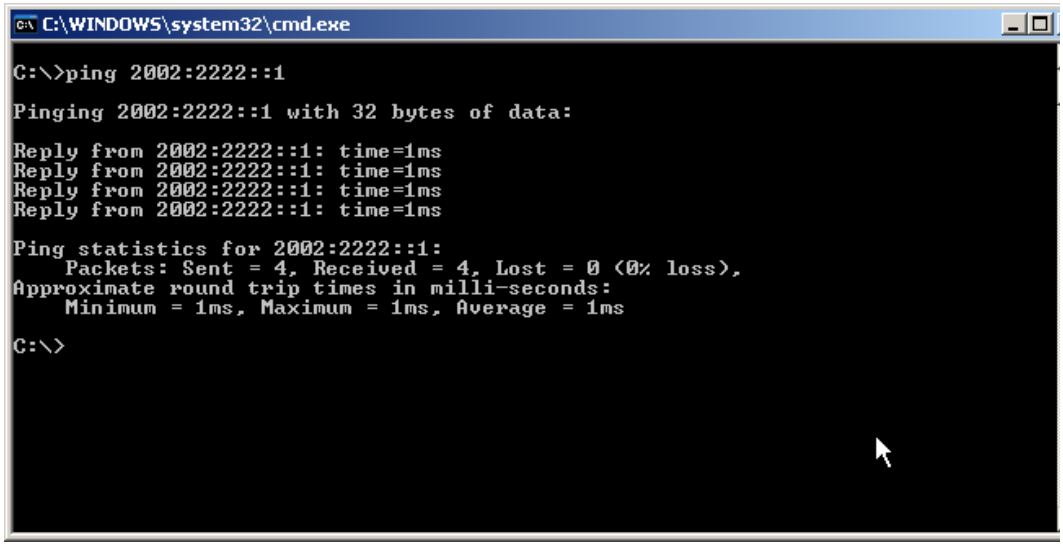
On Windows XP, you can view the PC's IPv6 information with the **ipv6 if** command. This displays IPv6 information for all network interfaces on the PC. Once you know the interface index number, you can specify it to view IPv6 information for that interface only:



To check if the PC can route to another IPv6 network, configure an IPv6 address on the second VLAN on the switch (VLAN2):

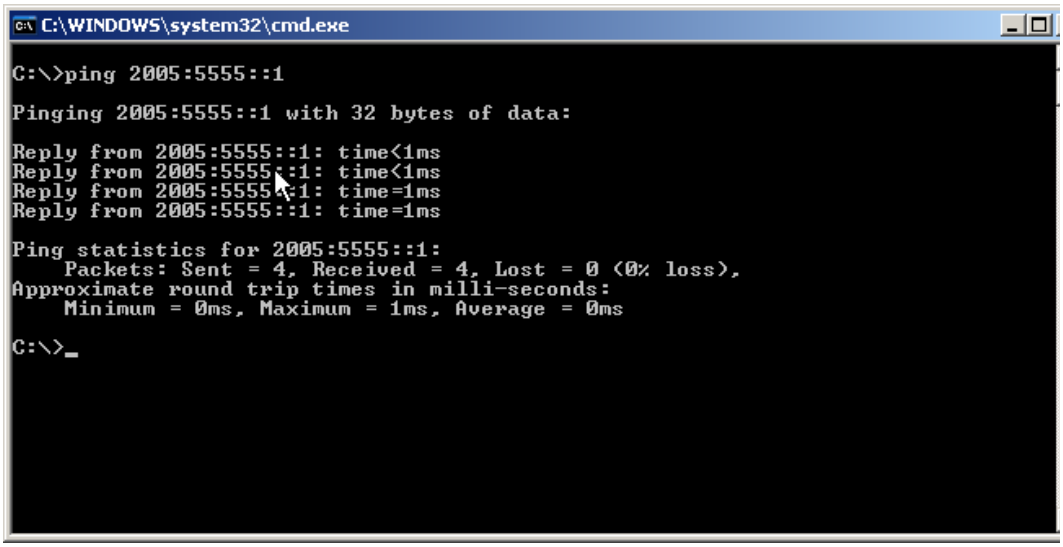
```
awplus(config-if)# ipv6 address 2002:2222::1/64
```

A ping to this address from the PC confirms that routing is functioning:



The following tests the ability to learn an IPv6 route (2005:5555::/64) via OSPFv3 on the switch, and checks connectivity to this from the PC:

```
awplus#sh ipv6 route
IPv6 Routing Table
Codes: C - connected, S - static, R - RIP, O - OSPF, B - BGP
Timers: Uptime

C   2001:1111::/64 via ::, vlan1, 01:20:48
C   2002:2222::/64 via ::, vlan2, 00:00:45
O   2005:5555::/64 [110/20] via fe80::eecd:6dff:fe20:c0e1, vlan2,
00:00:02
C   fe80::/64 via ::, vlan2, 00:00:45
C   fe80::/64 via ::, vlan1, 01:20:48
```



Finally, the following shows that a Telnet connection from the PC to the switch at 2005:5555::1/64 was successful:

```
Authenticator#show user
Line    User           Host(s)  Idle     Location       Priv Idletime
Timeout
vty 1   manager       idle    00:00:00  2001:1111::8f:3845:23e9:6fc515
10


Complete switch configuration:

!
vlan database
 vlan 2-3 state enable
!
interface port1.0.1
 switchport
 switchport mode access
!
interface port1.0.2
 switchport
 switchport mode access
 switchport access vlan 2
!
interface vlan1
 ip address 192.168.1.2/24
 ipv6 address 2001:1111::1/64
 no ipv6 nd suppress-ra
 ipv6 nd prefix 2001:1111::/64
!
interface vlan2
 ip address 192.168.2.1/24
 ipv6 address 2002:2222::1/64
!
ipv6 forwarding
!
```

Stopping multicast going to the CPU with L2 and L3 multicast configured

If both L2 and L3 multicast are being performed by a switch, unregistered multicast traffic arriving on ports in a VLAN that is only performing L2 multicasting (i.e. a VLAN on which PIM is not configured)will be sent to the CPU.

This is because although IGMP snooping will install an entry to stop the traffic being sent to the CPU by the L2 multicast process, the packets will also be passed to the L3 multicast process, which will send the packets to the CPU.

The **no multicast** command will stop the unregistered multicast from going to the CPU, by preventing the packets from being sent to the L3 multcasting process. The command is applied to a port which has multicast traffic arriving on it (the command can be applied to all ports in the L2 VLAN):

In the example below, the multicast video is arriving on port1.0.1

```
ip multicast-routing
!
vlan database
 vlan 1000,2000 state enable
!
interface port1.0.1
 switchport
switchport access vlan 2000
no multicast
!
interface port1.0.2
 switchport
switchport access vlan 2000
!
interface port1.0.3
 switchport
switchport access vlan 1000
!
interface vlan1000
 ip address 192.168.1.1/24
 ip pim dense-mode
!
interface vlan2000
 ip address 192.168.2.1/24
 ip igmp
 ip igmp version 2
!
```

Note: Even with this configuration, multicast traffic arriving on port1.0.1 will be correctly Layer 2 forwarded.

How to mitigate gateway forwarding issues with VRRPv2/3

This applies to v**5.4.3** and higher.

The issue

When a VRRP router either recovers or joins a VRRP gateway group and it has a higher priority than the current Master, by default it will preempt and take over as Master. If the upstream interface uses DHCP or stateless autoconfiguration, or the router itself uses a routing protocol for upstream connectivity, there may be a delay before it has the routing information to forward packets upstream.

VRRP by default preempts with no delay, resulting in a very fast switch over and moving to Master state. The only problem is that if the VRRP router is not ready to forward internal traffic upstream, then no packets can be routed.

How to mitigate this issue

Two methods are defined in this note, describing different approaches which may resolve the issue.

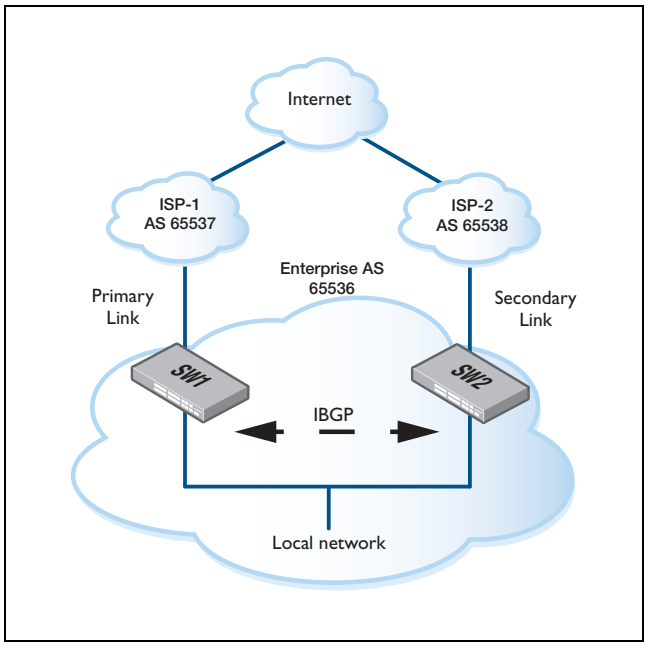
Method #1: Floating Static / Default Routes

This method involves creating static or default route(s) with a higher AD value than the AD of the dynamic routing protocol used for the upstream network.

Method #2: Disabling Preemption

This method involves disabling the automatic preemption of a VRRP router when it joins a VRRP gateway group. By disabling preemption, the device will not automatically re-elect itself as the VRRP Master if it has a higher priority.

This way the current VRRP Master will continue to forward traffic as normal and there will be no network disruption. If the Master fails, then the device will transition from Backup to Master as normal and in most cases the device will have learned all route forwarding information by then.



Configurations

SW1 Configuration:

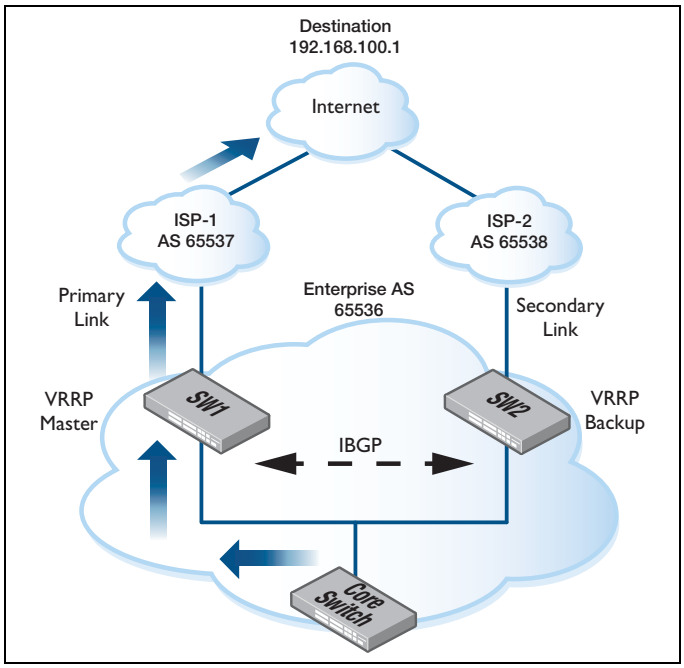
```
SW1(config)#vlan database
SW1(config-vlan)#vlan 10 name ISP-1
SW1(config-vlan)#vlan 20 name LAN

SW1(config)#interface vlan10
SW1(config-if)#description ISP-1
SW1(config-if)#ip address 192.168.10.1

SW1(config)#interface vlan20
SW1(config-if)#description LAN
SW1(config-if)#ip address 192.168.20.2

SW1(config)#interface port1.0.11
SW1(config-if)#switchport access vlan 10

SW1(config)#interface port1.0.12
SW1(config-if)#switchport access vlan 20

SW1(config)#ip prefix-list PERMIT_OUT_LIST seq 10 permit 192.168.20.0/24

SW1(config)#route-map PERMIT_OUT_MAP permit 10
SW1(config-route-map)#match ip address prefix-list PERMIT_OUT_LIST
SW1(config-route-map)#route-map PERMIT_OUT_MAP deny 20

SW1(config)#bgp extended-asn-cap

SW1(config)#router bgp 65536
SW1(config-router)#bgp router-id 1.1.1.1
SW1(config-router)#network 192.168.20.0/24
SW1(config-router)#neighbor 192.168.10.2 remote-as 65537
SW1(config-router)#neighbor 192.168.10.2 route-map PERMIT_OUT_MAP out
SW1(config-router)#neighbor 192.168.20.3 remote-as 65536
SW1(config-router)#neighbor 192.168.20.3 next-hop-self

SW1(config)#router vrrp 1 vlan20
SW1(config-router)#virtual-ip 192.168.20.1 backup
SW1(config-router)#priority 150
SW1(config-router)#enable
```

SW2 Configuration:

```
SW2(config)#vlan database
SW2(config-vlan)#vlan 10 name ISP-2
SW2(config-vlan)#vlan 20 name LAN

SW2(config)#interface vlan30
SW2(config-if)#description ISP-2
SW2(config-if)#ip address 192.168.30.1

SW2(config)#interface vlan20
SW2(config-if)#description LAN
SW2(config-if)#ip address 192.168.20.3

SW2(config)#interface port1.0.11
SW2(config-if)#switchport access vlan 30

SW2(config)#interface port1.0.12
SW2(config-if)#switchport access vlan 20

SW2(config)#bgp extended-asn-cap

SW2(config)#router bgp 65536
SW2(config-router)#bgp router-id 2.2.2.2
SW2(config-router)#network 192.168.20.0/24
SW2(config-router)#neighbor 192.168.20.2 remote-as 65536
SW2(config-router)#neighbor 192.168.20.2 next-hop-self
SW2(config-router)#neighbor 192.168.30.2 remote-as 65538
SW2(config-router)#neighbor 192.168.30.2 route-map PERMIT_OUT_MAP out

SW2(config)#ip prefix-list PERMIT_OUT_LIST seq 10 permit 192.168.20.0/24
SW2(config)#route-map PERMIT_OUT_MAP permit 10
SW2(config-route-map)#match ip address prefix-list PERMIT_OUT_LIST
SW2(config-route-map)#route-map PERMIT_OUT_MAP deny 20

SW2(config)#router vrrp 1 vlan20
SW2(config-router)#virtual-ip 192.168.20.1 backup
SW2(config-router)#priority 100
SW2(config-router)#enable
```

Example of the issue:

- Preemption is on by default.
- SW1 and SW2 use BGP to learn prefixes from the ISPs.
- Advertisement timers are at the default of 1 second (100 centiseconds) so a small amount of packet loss is expected.




```
SW1#show vrrp
VMAC enabled
Address family IPv4
VRRP Id: 1 on interface: vlan20
 State: AdminUp - Master
 Virtual IP address: 192.168.20.1 (Not-owner)
 Priority is 150
 Advertisement interval: 1 sec
 Preempt mode: TRUE
 Multicast membership on IPv4 interface vlan20: JOINED


SW2#show vrrp
VMAC enabled
Address family IPv4
VRRP Id: 1 on interface: vlan20
 State: AdminUp - Backup
 Virtual IP address: 192.168.20.1 (Not-owner)
 Priority is 100
 Advertisement interval: 1 sec
 Preempt mode: TRUE
 Multicast membership on IPv4 interface vlan20: JOINED
```

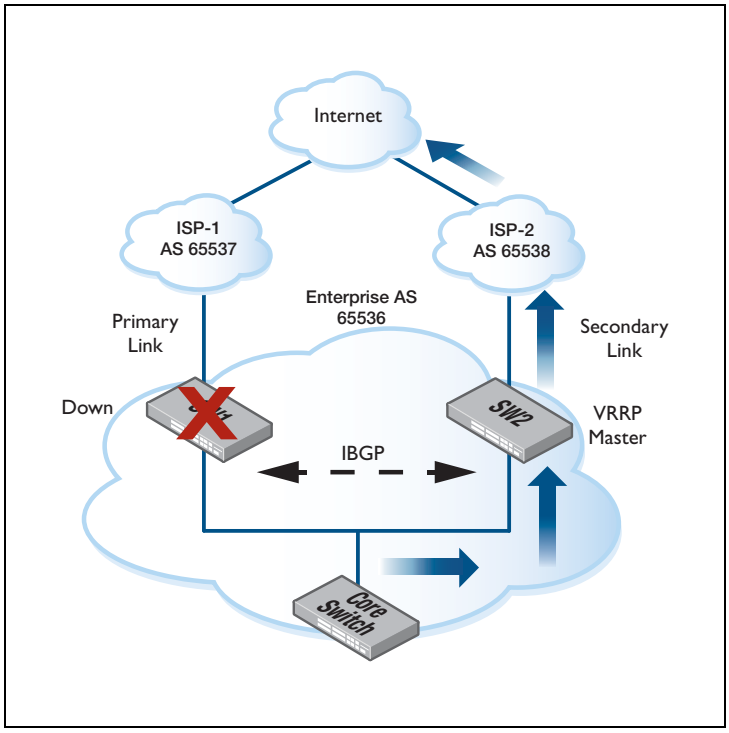
- Start a ping, from a switch on the LAN side of the VRRP routers, out towards the WAN side.

```
Core_Switch#ping 192.168.100.1 repeat 120
PING 192.168.100.1 (192.168.100.1) 56(84) bytes of data.
64 bytes from 192.168.100.1: icmp_req=1 ttl=63 time=1.31 ms
64 bytes from 192.168.100.1: icmp_req=2 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=3 ttl=63 time=2.27 ms
```

- Now, reboot the VRRP master.

```
SW1#reload
reboot system? (y/n): y
```

- Fairly quickly, the VRRP backup will transition to master.

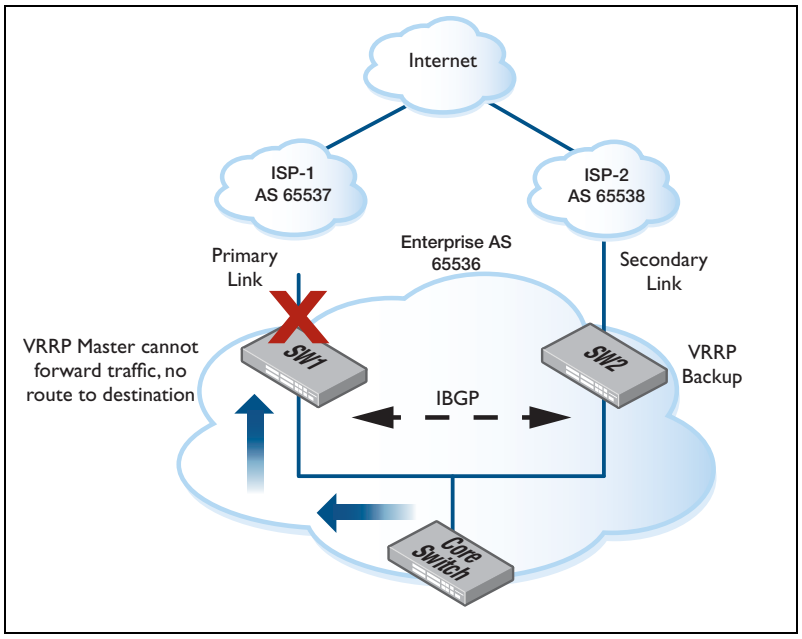



```
SW2#show vrrp
VMAC enabled
Address family IPv4
VRRP Id: 1 on interface: vlan20
 State: AdminUp - Master
 Virtual IP address: 192.168.20.1 (Not-owner)
 Priority is 100
 Advertisement interval: 1 sec
 Preempt mode: TRUE
 Multicast membership on IPv4 interface vlan20: JOINED
```

- SW2 is now the new Master.

```
64 bytes from 192.168.100.1: icmp_req=7 ttl=63 time=1.20 ms
!- Several packets were lost during the switchover at default
timers.

!- Notice the previous ICMP request number was 3 before
rebooting the VRRP master and is now 7.
64 bytes from 192.168.100.1: icmp_req=8 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=9 ttl=63 time=1.17 ms
64 bytes from 192.168.100.1: icmp_req=10 ttl=63 time=1.20 ms
64 bytes from 192.168.100.1: icmp_req=11 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=12 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=13 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=14 ttl=63 time=1.17 ms
64 bytes from 192.168.100.1: icmp_req=15 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=16 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=17 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=18 ttl=63 time=1.06 ms
64 bytes from 192.168.100.1: icmp_req=19 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=20 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=21 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=22 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=23 ttl=63 time=1.24 ms
64 bytes from 192.168.100.1: icmp_req=24 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=25 ttl=63 time=1.27 ms
64 bytes from 192.168.100.1: icmp_req=26 ttl=63 time=1.26 ms
64 bytes from 192.168.100.1: icmp_req=27 ttl=63 time=1.31 ms
64 bytes from 192.168.100.1: icmp_req=28 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=29 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=30 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=31 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=32 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=33 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=34 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=35 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=36 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=37 ttl=63 time=1.20 ms
64 bytes from 192.168.100.1: icmp_req=38 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=39 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=40 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=41 ttl=63 time=1.23 ms
64 bytes from 192.168.100.1: icmp_req=42 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=43 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=44 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=45 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=46 ttl=63 time=1.18 ms
64 bytes from 192.168.100.1: icmp_req=47 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=48 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=49 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=50 ttl=63 time=1.24 ms
64 bytes from 192.168.100.1: icmp_req=51 ttl=63 time=1.27 ms
64 bytes from 192.168.100.1: icmp_req=52 ttl=63 time=1.24 ms
64 bytes from 192.168.100.1: icmp_req=53 ttl=63 time=1.30 ms
```

```
64 bytes from 192.168.100.1: icmp_req=54 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=55 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=56 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=57 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=58 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=59 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=60 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=61 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=62 ttl=63 time=1.23 ms
64 bytes from 192.168.100.1: icmp_req=63 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=64 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=65 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=66 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=67 ttl=63 time=1.20 ms
64 bytes from 192.168.100.1: icmp_req=68 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=69 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=70 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=71 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=72 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=73 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=74 ttl=63 time=1.24 ms
64 bytes from 192.168.100.1: icmp_req=75 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=76 ttl=63 time=1.23 ms
64 bytes from 192.168.100.1: icmp_req=77 ttl=63 time=1.27 ms
64 bytes from 192.168.100.1: icmp_req=78 ttl=63 time=1.30 ms
64 bytes from 192.168.100.1: icmp_req=79 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=80 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=81 ttl=63 time=1.23 ms
64 bytes from 192.168.100.1: icmp_req=82 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=83 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=84 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=85 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=86 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=87 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=88 ttl=63 time=1.20 ms
64 bytes from 192.168.100.1: icmp_req=89 ttl=63 time=1.23 ms
64 bytes from 192.168.100.1: icmp_req=90 ttl=63 time=1.18 ms
64 bytes from 192.168.100.1: icmp_req=91 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=92 ttl=63 time=1.21 ms

!At this point, the VRRP master has completed reboot, and resumes
its role as VRRP master.
From 192.168.20.2 icmp_seq=93 Destination Net Unreachable
From 192.168.20.2 icmp_seq=94 Destination Net Unreachable
From 192.168.20.2 icmp_seq=95 Destination Net Unreachable
From 192.168.20.2 icmp_seq=96 Destination Net Unreachable

!Due to the VRRP router (SW1) preempting and assuming its place
as Master before it has learned the default route from the BGP
peers,it is unable to forward packets to the destination.
```



- Check the route table on the returned VRRP master.

```
SW1#show ip route
Codes: C - connected, S - static, R - RIP, B - BGP
       O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       * - candidate default
C        10.36.4.0/24 is directly connected, vlan1000
C        192.168.10.0/24 is directly connected, vlan10
C        192.168.20.0/24 is directly connected, vlan20
```

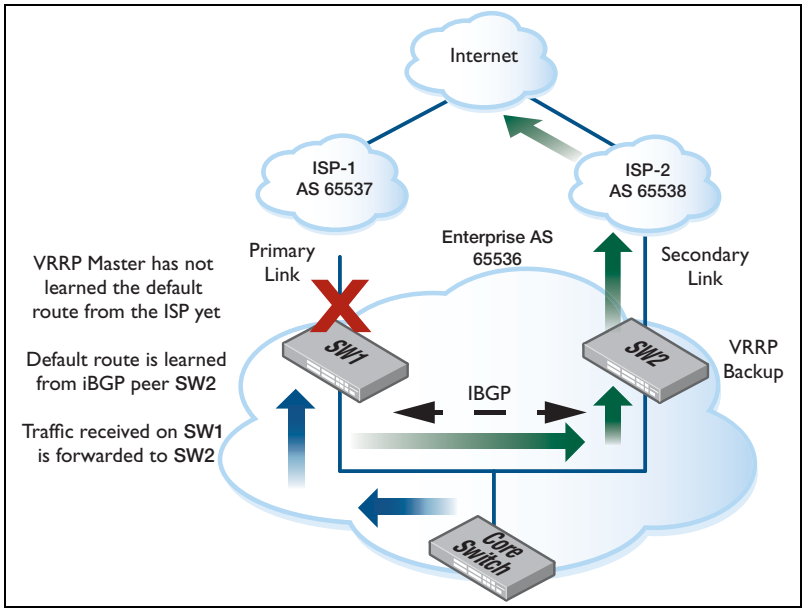
- SW1 has not established BGP peering sessions yet and has not learnt the default route.

```
SW1#show vrrp
VMAC enabled
Address family IPv4
VRRP Id: 1 on interface: vlan20
 State: AdminUp - Master
 Virtual IP address: 192.168.20.1 (Not-owner)
 Priority is 150
 Advertisement interval: 1 sec
 Preempt mode: TRUE
 Multicast membership on IPv4 interface vlan20: JOINED
```

- SW1 is the VRRP Master, so packets have been forwarded to the device.
- Eventually, the pings start to succeed again.

```
64 bytes from 192.168.100.1: icmp_req=97 ttl=63 time=1.14 ms
64 bytes from 192.168.100.1: icmp_req=98 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=99 ttl=63 time=1.24 ms
```

- Check the route table on the VRRP master again.

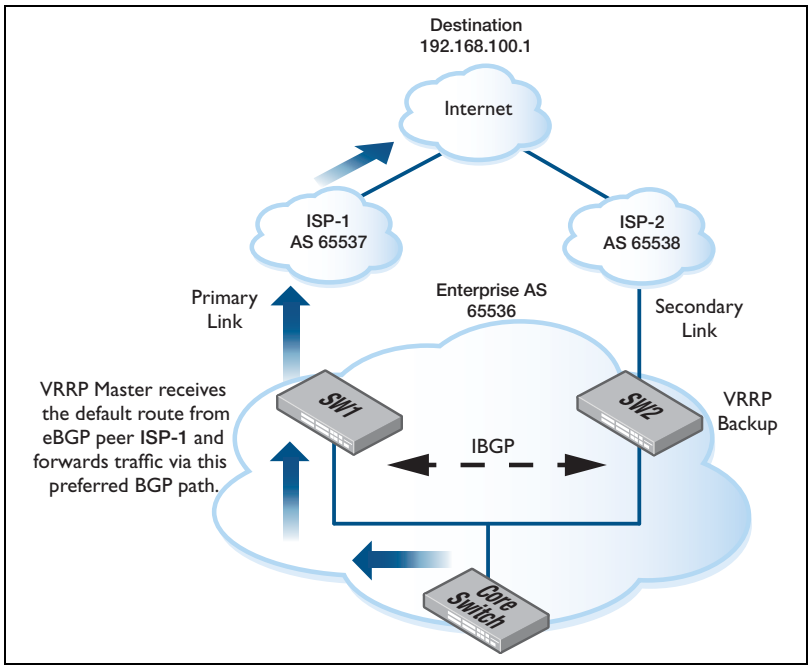



```
SW1#show ip route
Codes: C - connected, S - static, R - RIP, B - BGP
O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2
* - candidate default
Gateway of last resort is 192.168.20.3 to network 0.0.0.0
B* 0.0.0.0/0 [200/0] via 192.168.20.3, vlan20, 00:00:04
C 10.36.4.0/24 is directly connected, vlan1000
C 192.168.10.0/24 is directly connected, vlan10
C 192.168.20.0/24 is directly connected, vlan20
```

- The default route has been learned from the iBGP peer first due to a shorter route advertisement timer.

Note: iBGP route advertisement interval is 5 seconds. eBGP route advertisement interval is 30 seconds.

```
64 bytes from 192.168.100.1: icmp_req=100 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=101 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=102 ttl=63 time=1.24 ms
64 bytes from 192.168.100.1: icmp_req=103 ttl=63 time=1.28 ms
64 bytes from 192.168.100.1: icmp_req=104 ttl=63 time=1.31 ms
64 bytes from 192.168.100.1: icmp_req=105 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=106 ttl=63 time=1.23 ms
64 bytes from 192.168.100.1: icmp_req=107 ttl=63 time=1.23 ms
64 bytes from 192.168.100.1: icmp_req=108 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=109 ttl=63 time=1.23 ms
64 bytes from 192.168.100.1: icmp_req=110 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=111 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=112 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=113 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=114 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=115 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=116 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=117 ttl=63 time=1.21 ms
64 bytes from 192.168.100.1: icmp_req=118 ttl=63 time=1.23 ms
64 bytes from 192.168.100.1: icmp_req=119 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=120 ttl=63 time=1.21 ms

--- 192.168.100.1 ping statistics ---
120 packets transmitted, 113 received, +4 errors, 5% packet loss, time
119115ms
rtt min/avg/max/mdev = 1.065/1.232/2.277/0.113 ms

SW1#show ip route
Codes: C - connected, S - static, R - RIP, B - BGP
       O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       * - candidate default
Gateway of last resort is 192.168.10.2 to network 0.0.0.0

B*        0.0.0.0/0 [20/0] via 192.168.10.2, vlan10, 00:04:02
C         10.36.4.0/24 is directly connected, vlan1000
C         192.168.10.0/24 is directly connected, vlan10
C         192.168.20.0/24 is directly connected, vlan20

!- Once the default route is learned from the external BGP peer,
it is the preferred path.
```



Issue analysis:

SW1 finished rebooting and preempted causing it to assume its place as the VRRP Master. This then caused SW2 to re-assume its position as VRRP Backup. Traffic was then forwarded to SW1, but SW1 had not received the default route from either BGP peer yet. If there was no iBGP peering, the time taken to learn the default route from the external peer would be much longer, resulting in additional downtime.

While the iBGP peer advertises the default route first, (preventing additional downtime) this then causes inefficient routing where traffic received by the VRRP Master must be transmitted back onto the link it was received and forwarded to the VRRP Backup. This behavior ensures until the default route is learned via the external BGP peer.

Implementing mitigation method #1

- Floating default routes are created on SW1 and SW2.

This has a higher AD than eBGP and iBGP routes (20 and 200 respectively). As soon as a BGP route is learned, it will be preferred over the floating default route.

```
SW1(config)#ip route 0.0.0.0/0 192.168.10.2 250
SW2(config)#ip route 0.0.0.0/0 192.168.30.2 250
```

Testing method #1:

```
64 bytes from 192.168.100.1: icmp_req=88 ttl=63 time=1.23 ms
64 bytes from 192.168.100.1: icmp_req=89 ttl=63 time=1.23 ms
64 bytes from 192.168.100.1: icmp_req=90 ttl=63 time=1.23 ms
64 bytes from 192.168.100.1: icmp_req=91 ttl=63 time=1.24 ms

64 bytes from 192.168.100.1: icmp_req=93 ttl=63 time=3.66 ms
!- Several packets lost during the switch over due to the default timers
used, but Master was able to forward immediately.

64 bytes from 192.168.100.1: icmp_req=95 ttl=63 time=1.22 ms
64 bytes from 192.168.100.1: icmp_req=96 ttl=63 time=1.20 ms
64 bytes from 192.168.100.1: icmp_req=97 ttl=63 time=1.19 ms
64 bytes from 192.168.100.1: icmp_req=98 ttl=63 time=1.21 ms
```

- Floating default route is installed immediately.

```
SW1#show ip route
Codes: C - connected, S - static, R - RIP, B - BGP
       O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       * - candidate default

Gateway of last resort is 192.168.10.2 to network 0.0.0.0

S* 0.0.0.0/0 [250/0] via 192.168.10.2, vlan10
C 10.36.4.0/24 is directly connected, vlan1000
C 192.168.10.0/24 is directly connected, vlan10
C 192.168.20.0/24 is directly connected, vlan20
```

- SW1 is the VRRP Master.

```
SW1#show vrrp
VMAC enabled
Address family IPv4
VRRP Id: 1 on interface: vlan20
State: AdminUp - Master
Virtual IP address: 192.168.20.1 (Not-owner)
Priority is 150
Advertisement interval: 1 sec
Preempt mode: TRUE
Multicast membership on IPv4 interface vlan20: JOINED
```

- The default route from iBGP peer is learned and has a lower AD.

```
SW1#show ip route
Codes: C - connected, S - static, R - RIP, B - BGP
       O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       * - candidate default

Gateway of last resort is 192.168.20.3 to network 0.0.0.0

B*      0.0.0.0/0 [200/0] via 192.168.20.3, vlan20, 00:00:12
C       10.36.4.0/24 is directly connected, vlan1000
C       192.168.10.0/24 is directly connected, vlan10
C       192.168.20.0/24 is directly connected, vlan20
```

- The default route from eBGP peer is learned with a lower AD than the internal route.

```
SW1#show ip route
Codes: C - connected, S - static, R - RIP, B - BGP
       O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       * - candidate default
Gateway of last resort is 192.168.10.2 to network 0.0.0.0
B*      0.0.0.0/0 [20/0] via 192.168.10.2, vlan10, 00:00:02
C       10.36.4.0/24 is directly connected, vlan1000
C       192.168.10.0/24 is directly connected, vlan10
C       192.168.20.0/24 is directly connected, vlan20
```

Implementing mitigation method #2:

This method does not require a display of the test as SW2 will continue to forward traffic as SW1 will not preempt to re-assume its position as Master.

```
SW1(config)#router vrrp 1 vlan20
SW1(config-router)#disable
SW1(config-router)#preempt-mode false
SW1(config-router)#enable
SW2(config)#router vrrp 1 vlan20
SW2(config-router)#disable
SW2(config-router)#preempt-mode false
SW2(config-router)#enable
```

Verifying mitigation method #2:

Part 1: SW1 is the VRRP Master - Preemption is disabled on each device.

```
SW1#show vrrp
VMAC enabled
Address family IPv4
VRRP Id: 1 on interface: vlan20
 State: AdminUp - Master
 Virtual IP address: 192.168.20.1 (Not-owner)
 Priority is 150
 Advertisement interval: 1 sec
 Preempt mode: FALSE
 Multicast membership on IPv4 interface vlan20: JOINED

SW2#show vrrp
VMAC enabled
Address family IPv4
VRRP Id: 1 on interface: vlan20
 State: AdminUp - Backup
 Virtual IP address: 192.168.20.1 (Not-owner)
 Priority is 100
 Advertisement interval: 1 sec
 Preempt mode: FALSE
 Multicast membership on IPv4 interface vlan20: JOINED
```

Part 2: Switchover

- SW1 is rebooted.
- SW2 becomes the VRRP Master.

```
SW1#reload
reboot system? (y/n): y

SW2#show vrrp
VMAC enabled
Address family IPv4
VRRP Id: 1 on interface: vlan20
State: AdminUp - Master
Virtual IP address: 192.168.20.1 (Not-owner)
Priority is 100
Advertisement interval: 1 sec
Preempt mode: FALSE
Multicast membership on IPv4 interface vlan20: JOINED
```

Part 3: Verify

- SW1 finishes rebooting.
- SW1 does not preempt, and assumes its position as VRRP Backup.
- SW2 remains as the VRRP Master.

```
SW1#show vrrp
VMAC enabled
Address family IPv4
VRRP Id: 1 on interface: vlan20
State: AdminUp - Backup
Virtual IP address: 192.168.20.1 (Not-owner)
Priority is 150
Advertisement interval: 1 sec
Preempt mode: FALSE
Multicast membership on IPv4 interface vlan20: JOINED

SW2#show vrrp
VMAC enabled
Address family IPv4
VRRP Id: 1 on interface: vlan20
State: AdminUp - Master
Virtual IP address: 192.168.20.1 (Not-owner)
Priority is 100
Advertisement interval: 1 sec
Preempt mode: FALSE
Multicast membership on IPv4 interface vlan20: JOINED
```

How to mitigate flooding of VRRP advertisements on the LAN

Versions this applies to v**5.4.3** and higher.

Issue:

By default VRRPv2 and VRRPv3 Master routers multicast ADVERTISEMENT messages to 224.0.0.18 (v2 and v3) and FF02::12 (v3 only). Because these are multicast, IPv4 and IPv6 multicast Ethernet addresses are used when forwarding the frame out onto the link.

RFC4541 (section 3, IPv6 Considerations) states that MLD messages are not sent regarding groups with addresses in the FF00::/15 range. Similarly, IGMP messages should not be sent for the reserved IPv4 multicast groups 224.0.0.x. This means that IGMP and MLD snooping are unable to detect the ports to which to forward VRRP multicast frames.

The end result is the VRRP multicast Ethernet frame is flooded out all ports as unregistered multicast packets.

VRRP is very chatty due to the requirement for fast switchover of gateways.

In Layer 2 networks where VRRPv2/3 routers are deployed, users may encounter situations where VRRP messages are flooded to links where no VRRP routers reside. If VRRP is tuned for very high availability (eg less than 10 centiseconds) and multiple VRRP instances have been configured, this could result in many VRRP ADVERTISEMENT messages sent per VRRP Master, undesirably being flooded down throughout the entire Layer 2 network, causing unnecessary link utilisation.

Fortunately, there is a method for mitigating this problem.

When to use this mitigation method:

- When VRRP ADVERTISEMENT messages are flooded unnecessarily to areas of the network where VRRP routers do not reside.
- When VRRP ADVERTISEMENT interval is very low, resulting in many packets being sent per second,which may result in unnecessary chatty links where VRRP routers do not reside.

When not to use this mitigation method:

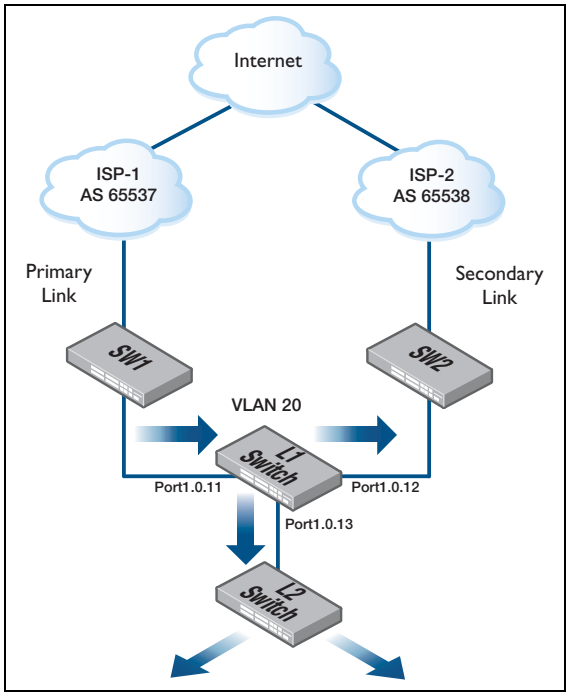
- If the network often has VRRP routers added or removed from the topology.
- If VRRP routers may be added to the VLAN at any time and a static setup is not appropriate.

More Information:

- http://tools.ietf.org/search/rfc4541 – Considerations for IGMP and MLD Snooping switches
- http://tools.ietf.org/search/rfc3768 – VRRPv2
- http://tools.ietf.org/search/rfc5798 – VRRPv3 (for IPv4 and IPv6)

Default VRRP Setup:

- SW1 sends VRRP ADVERTISEMENT messages to 224.0.0.18 and FF02::12 respectively.
- The Layer 2 switch connecting SW1 and SW2 does not perform snooping for the reserved multicast ranges, as per the RFC standard.
- The Layer 2 switch then floods the frame out all ports due to there being no matching entry. Subsequent Layer 2 switches in the topology also flood the frame.
- The result is unnecessary frame transmissions on internal network links.



Mitigation Method:

Issue as seen on the L2 switch connecting the VRRP routers and the rest of the network:

- This example shows the port counters incrementing as VRRP ADVERTISEMENT messages are sent out switchports where no VRRP routers reside.
- Advertisement interval is set very low for both IPv4 and IPv6 gateway groups, resulting in many packets being sent per second.

Interface connected to VRRPv3 Master (for IPv4 and IPv6 groups)

```
L2_Switch#show interface port1.0.11
Interface port1.0.11
  Scope: both
  Link is UP, administrative state is UP
  Thrash-limiting
    Status Not Detected, Action learn-disable, Timeout 1(s)
  Hardware is Ethernet, address is eccd.6d20.c073
  index 5011 metric 1 mru 1500
  current duplex full, current speed 1000, current polarity mdix
  configured duplex auto, configured speed auto, configured polarity auto
<UP,BROADCAST,RUNNING,MULTICAST>
SNMP link-status traps: Disabled
  input packets 1760, bytes 128480, dropped 0, multicast packets 1760
  output packets 0, bytes 0, multicast packets 0 broadcast packets 0
Time since last state change: 0 days 00:26:22
```

Interface connected to VRRPv3 Backup

```
L2_Switch#show interface port1.0.12
Interface port1.0.12
  Scope: both
  Link is UP, administrative state is UP
  Thrash-limiting
   Status Not Detected, Action learn-disable, Timeout 1(s)
  Hardware is Ethernet, address is eccd.6d20.c073
  index 5012 metric 1 mru 1500
  current duplex full, current speed 1000, current polarity mdi
  configured duplex auto, configured speed auto, configured polarity auto
<UP,BROADCAST,RUNNING,MULTICAST>
SNMP link-status traps: Disabled
  input packets 0, bytes 0, dropped 0, multicast packets 0
  output packets 1802, bytes 131546, multicast packets 1802 broadcast packets 0
Time since last state change: 0 days 00:26:23
```

Interface connected to downstream L2 Switch

VRRP packets are needlessly being flooded out this port to the rest of the network.

```
L2_Switch#show interface port1.0.13
Interface port1.0.13
  Scope: both
  Link is UP, administrative state is UP
  Thrash-limiting
   Status Not Detected, Action learn-disable, Timeout 1(s)
  Hardware is Ethernet, address is eccd.6d20.c073
  index 5013 metric 1 mru 1500
  current duplex full, current speed 1000, current polarity mdix
  configured duplex auto, configured speed auto, configured polarity auto
  <UP,BROADCAST,RUNNING,MULTICAST>
  SNMP link-status traps: Disabled
   input packets 0, bytes 0, dropped 0, multicast packets 0
  output packets 1842, bytes 134466, multicast packets 1842 broadcast packets 0
 Time since last state change: 0 days 00:26:24
```

Mitigation Method:

While the RFC standard does not allow IGMP or MLD reports for the reserved addresses, we may still execute administrative control to prevent unnecessary flooding of the VRRP multicast messages.

Apply Static Group Membership:

All ports connected to VRRP routers must be specified for the relevant multicast group.

VRRPv2 / VRRPv3 IPv4:

```
L2_Switch(config)#interface vlan20
L2_Switch(config-if)#ip igmp static-group 224.0.0.18 interface
   port1.0.11
L2_Switch(config-if)#ip igmp static-group 224.0.0.18 interface
   port1.0.12
```

VRRPv3 IPv6:

```
L2_Switch(config)#interface vlan20
L2_Switch(config-if)#ipv6 mld static-group ff02::12 interface
   port1.0.11
L2_Switch(config-if)#ipv6 mld static-group ff02::12 interface
   port1.0.12
```

- This will have the effect that the multicast groups 224.0.0.18 and ff02::12 are no longer unregistered.
- The switch will no longer flood these groups, but just forward them to the ports which have been specified in the static forwarding commands.

The prevention of the flooding can be seen by looking at port counters.

Clear the port counters:

(This assists in confirming no packets are being flooded out the port)

```
L2_Switch#clear port counter
```

Check the port counters again for input and output multicast packet:

- Interface connected to VRRPv3 Master (for IPv4 and IPv6 groups).

```
L2_Switch#show interface port1.0.11
Interface port1.0.11
  Scope: both
  Link is UP, administrative state is UP
  Thrash-limiting
   Status Not Detected, Action learn-disable, Timeout 1(s)
  Hardware is Ethernet, address is eccd.6d20.c073
  index 5011 metric 1 mru 1500
  current duplex full, current speed 1000, current polarity mdix
  configured duplex auto, configured speed auto, configured polarity auto
  <UP,BROADCAST,RUNNING,MULTICAST>
  SNMP link-status traps: Disabled
    input packets 550, bytes 40150, dropped 0, multicast packets 550
    output packets 0, bytes 0, multicast packets 0 broadcast packets 0
  Time since last state change: 0 days 00:31:30
```

- Interface connected to VRRPv3 Backup.

```
L2_Switch#show interface port1.0.12
Interface port1.0.12
  Scope: both
  Link is UP, administrative state is UP
  Thrash-limiting
   Status Not Detected, Action learn-disable, Timeout 1(s)
  Hardware is Ethernet, address is eccd.6d20.c073
  index 5012 metric 1 mru 1500
  current duplex full, current speed 1000, current polarity mdi
  configured duplex auto, configured speed auto, configured polarity auto
  <UP,BROADCAST,RUNNING,MULTICAST>
  SNMP link-status traps: Disabled
    input packets 0, bytes 0, dropped 0, multicast packets 0
    output packets 550, bytes 40150, multicast packets 550 broadcast packets 0
  Time since last state change: 0 days 00:31:31
```

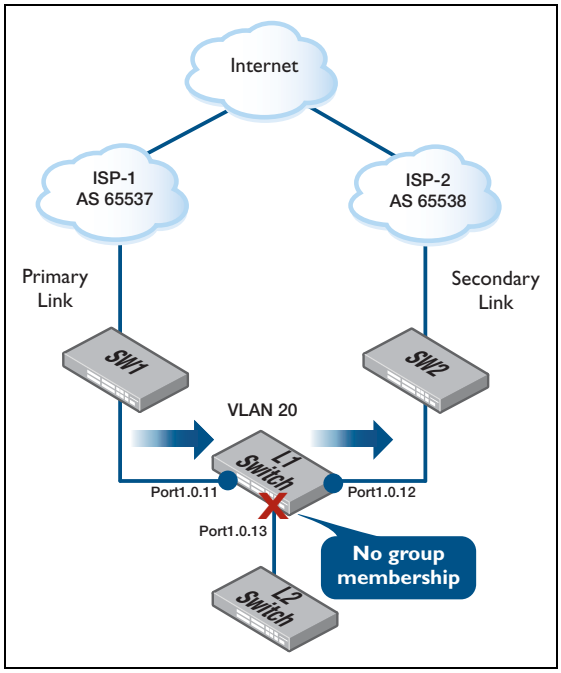
- Interface connected to downstream L2 Switch.

No VRRP multicast packets are being sent out this interface.

```
L2_Switch#show interface port1.0.13
Interface port1.0.13
  Scope: both
  Link is UP, administrative state is UP
  Thrash-limiting
    Status Not Detected, Action learn-disable, Timeout 1(s)
  Hardware is Ethernet, address is eccd.6d20.c073
  index 5013 metric 1 mru 1500
  current duplex full, current speed 1000, current polarity mdix
  configured duplex auto, configured speed auto, configured polarity auto
  <UP,BROADCAST,RUNNING,MULTICAST>
  SNMP link-status traps: Disabled
    input packets 0, bytes 0, dropped 0, multicast packets 0
    output packets 0, bytes 0, multicast packets 0 broadcast packets 0
  Time since last state change: 0 days 00:31:35
```

Result:

Port1.0.11 and Port1.0.12 are now have a static group membership for 224.0.0.18 and FF02::12 multicast groups. VRRP ADVERTISEMENT messages will not be flooded out ports where VRRP routers do not exist. So, port 1.0.13 is not sending VRRP packets.



Considerations:

Care must be taken to configure a static group entry for all ports where a VRRP router is reachable for the VLAN. Failure to include a static membership for a port where a VRRP router resides, will result in no VRRP ADVERTISEMENT messages reaching that neighbor.

Expansion of the network must also be considered. If additional VRRP routers are added to the VLAN, reachable out different ports, then these ports must also have a static multicast group membership entry.

High CPU utilisation caused by Windows uPnP

Background

Microsoft Windows machines can use the Universal Plug and Play (UPnP) set of networking protocols to discover devices on the network. Part of the UPnP suite is SSDP (Simple Service Discovery Protocol), which is used to discover and advertise network services. The multicast address used by SSDP is 239.255.255.250.

Another protocol which uses multicast to announce services on a local network is SLP (Service Location Protocol). This uses the multicast address 239.255.255.253

The problem

If a switch has a number of VLANs configured, and has PIM configured on them, each time a packet with either of these multicast addresses arrives from a different source, the switch must add an entry for all downstream PIM interfaces. This operation is CPU intensive, and

can cause the CPU utilisation to be increased considerably if enough of this traffic is received. The output below was taken from a switch experiencing high CPU utilisation from this situation:

```
CPU averages:
 1 second: 71%, 20 seconds: 68%, 60 seconds: 70%
System load averages:
 1 minute: 2.18, 5 minutes: 1.91, 15 minutes: 1.44
Current CPU load:
 userspace: 73%, kernel: 19%, interrupts: 2% iowaits: 0%

user processes
==============
  pid name              thrds  cpu%   pri state sleep% runtime
 1531 pdmd                 1  42.7    20   run      0  9264133
 1467 nsm                  1  20.2    20 sleep      0  6488758
```

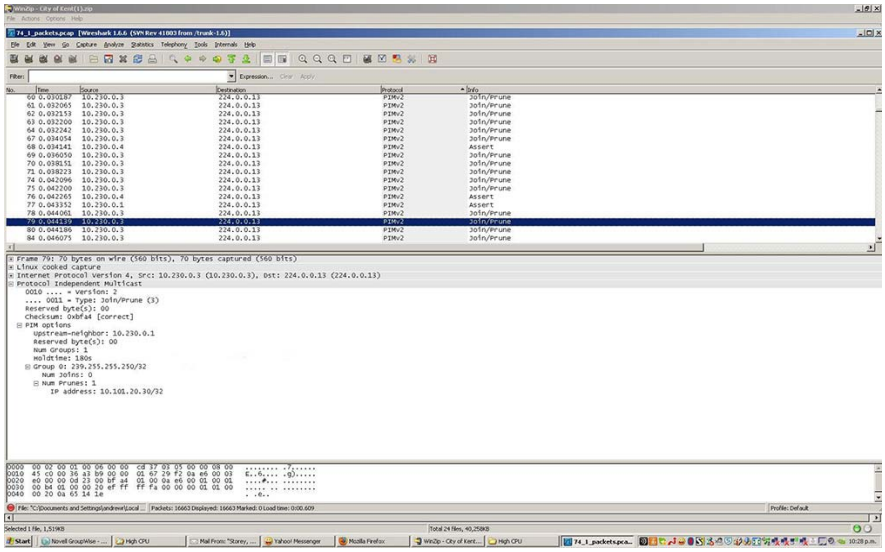
From this output, we can see that the two processes with the highest CPU utilisation are:

- The **pdmd** process is the PIM Dense-Mode module.
- The **nsm** process is the route table management module, which PIM has to keep telling what forwarding entries to create, delete, and update.

A capture of packets to/from the switch CPU, as seen below, shows a lot of PIM activity, all within a very short time period.The reason is that there are packets to 239.255.255.250 from many different sources. So you end up with several PIM trees, with different sources, that are all for the group 239.255.255.250.

Any IGMP Membership Reports (Joins) or Leaves for these multicast groups will cause increased CPU utilisation, as each of these many forwarding trees must be updated to reflect the effect of the IGMP signalling.

Also, traffic for 239.255.255.250 will often be turning up on multiple interfaces, which adds the complication that the traffic is arriving on interfaces that are not the RPF interface to other sources that are sending traffic to 239.255.255.250, which means that the switch has to send out asserts.



A clear indicator that the switch is receiving traffic to these group addresses from multiple sources is that the output of `show platform table ipmulti` will contain several entries for the groups 239.255.255.250 and/or 239.255.255.253, each from different source addresses, and possibly with different RPF interfaces, similar to the following:

```
2196 239.255.255.250 10.2.2.149      1  -- -- --   2   0  Ena   4    23  0
      Mll=658   VID=13   UseVidx=1 VIDX=4126  TTLThr=1    ExclSrcVlan=0
        Ports = port1.1.9

      Mll=658   VID=11   UseVidx=1 VIDX=4108  TTLThr=1    ExclSrcVlan=0
        Ports = port1.1.10, port1.1.6

      Mll=659   VID=1    UseVidx=1 VIDX=4100  TTLThr=1    ExclSrcVlan=0
        Ports = None

2692 239.255.255.250 10.2.2.156      1  -- -- --   2   0  Ena   4    23  0
      Mll=1350  VID=13   UseVidx=1 VIDX=4126  TTLThr=1    ExclSrcVlan=0
        Ports = port1.1.9

      Mll=1350  VID=11   UseVidx=1 VIDX=4108  TTLThr=1    ExclSrcVlan=0
        Ports = port1.1.10, port1.1.6

2340 239.255.255.250 10.2.3.7        1  -- -- --   2   0  Ena   4    31  1
      Mll=1388  VID=13   UseVidx=1 VIDX=4126  TTLThr=1    ExclSrcVlan=0
        Ports = port1.1.9

      Mll=1388  VID=11   UseVidx=1 VIDX=4108  TTLThr=1    ExclSrcVlan=0
        Ports = port1.1.10, port1.1.6

      Mll=1389  VID=1    UseVidx=1 VIDX=4100  TTLThr=1    ExclSrcVlan=0
        Ports = None


2156 239.255.255.250 10.2.3.10       1  -- -- --   2   0  Ena   4    31  1
      Mll=215   VID=13   UseVidx=1 VIDX=4126  TTLThr=1    ExclSrcVlan=0
        Ports = port1.1.9

      Mll=215   VID=11   UseVidx=1 VIDX=4108  TTLThr=1    ExclSrcVlan=0
        Ports = port1.1.10, port1.1.6

      Mll=220   VID=1    UseVidx=1 VIDX=4100  TTLThr=1    ExclSrcVlan=0
        Ports = None
```

Another problem that can occur on a switch receiving lots of this multicast traffic, is that it can fill up the L3 multicast table, and possibly prevent desired multicast entries from being accomodated in the table. An indicator that the table is being filled is that error messages will appear in the log i.e:

```
2014 Jan 28 17:52:04 local6.warning CoreSwitch EXFX[1767]: Unable to
add MC L3 route (GRP: 239.255.255.250/32, SRC: 10.2.1.147/32). Route
Entry table full
```

The resolution

To stop this multicast from reaching the switch's CPU, we can configure an ACL as follows:

```
access-list hardware acl-drop-SSDP_SLP
   deny udp any 239.255.255.250/32
```

```
   deny udp any 239.255.255.253/32
```

and apply it either globally, or on specific interfaces as required:

```
interface port1.1.1
  access-group acl-drop-SSDP_SLP
```

Blocking IGMPv3 Reports for these groups

Even when the ACL has been put in place to block the receipt of multicast streams destined to 239.255.255.250 and 239.255.255.253, it is still possible to find IGMP forwarding entries for these groups, and also find (*,g) entries for these groups in the **show platform table ipmulti** output.

For example, the **show IGMP Groups** output can continue to contain entries like:

```
239.255.255.250  vlan11              20:44:00 00:04:19 10.2.1.26
239.255.255.253  vlan11              20:44:01 00:04:19 10.2.1.21
239.255.255.250  vlan12              20:43:58 00:04:14 10.2.1.96
239.255.255.250  vlan13              17:16:40 00:04:18 10.2.1.141
239.255.255.253  vlan13              17:16:40 00:04:13 10.2.1.141
239.255.255.250  vlan14              19:47:14 00:04:08 10.2.1.202
239.255.255.253  vlan14              19:47:14 00:04:18 10.2.1.202
```

and, the **show platform table ipmulti** output can contain entries like:

```
105 239.255.255.253 0.0.0.0        1  -- -- --   2   1  Ena   6  4094   1
       -------------no downstream interfaces--------------


  72 239.255.255.250 0.0.0.0       1  -- -- --   2   1  Ena   6  4094   1
       -------------no downstream interfaces--------------
```

- The reason for this is that the ACL defined above as acl-drop-SSDP_SLP will not drop IGMPv3 reports for these groups. They will drop IGMPv2 reports for the groups 239.255.255.250 and 239.255.255.253 because the dest IP on IPGMPv2 reports is the group address.
- But, they will not drop IGMPv3 reports for the groups 239.255.255.250 and 239.255.255.253 ,as IGMPv3 joins have dest IP 224.0.0.22.
- So, if any devices in the network are sending in IGMPv3 reports for those groups, then the switch will create IGMP membership entries for those groups.
- But, that is not a big problem ,as it only means one table entry per group per VLAN, whereas the BIG problem was the numerous streams to 239.255.255.250 and 239.255.255.253, from numerous source addresses.

If you do also want to eliminate the IGMP group membership entries for the unwanted groups, you can use IGMP: filtering

```
access-list 1
  deny 239.255.255.250 0.0.0.0
  deny 239.255.255.253 0.0.0.0
  permit 224.0.0.0   15.255.255.255
exit

 interface vlan11
 ip igmp access-group 1
 interface vlan12
 ip igmp access-group 1
```

Resiliency

VLAN-based resiliency link

Introduction

The resiliency link is an important component in the AlliedWare Plus Virtual Chassis Stacking (VCStack™) solution.

The resiliency link is an extra link between the stack members, which is independent of the stacking connections. It is used when switches lose contact with each other over the stacking connection. This link allows the Backup Member switch(es) to determine if the master is still present, and operational, via health-check messages sent by the master over the resiliency link interface.

Without a resiliency link: if communication is lost over the stacking connection, a Backup Member will automatically transition to Master status. So, if the Master switch was still operational, there would now be two active Masters in the stack.

With a resiliency link: the Backup members can see if the Master is still operational, so no Backup member transitions to Master unless it is required.

On the SwitchBlade™ x908, and the x900 family of switches, the out-of-band Ethernet port functions as the resiliency link interface. However, other models of switch don't have an out-of-band Ethernet port. So a resiliency link within a VCStack of these other models must use a switch port or ports. Because healthcheck messages need to be received by each stack Backup member unit, this means giving up one or more front-panel ports per switch, to be used solely for resiliency-link purposes.

The solution - a resiliency VLAN

The switch port(s) that will function as the resiliency link should be assigned to a dedicated VCStack resiliency VLAN.

The resiliency VLAN should not be either:

- The Stack Management VLAN, or
- A VLAN that will carry any user traffic.

This VLAN must be used only for resiliency purposes, and should only carry data about VCStack healthcheck messages. This is achieved by not creating the resiliency VLAN in the switch's "VLAN Database" (like other user-defined VLANs).

There are two reasons for this:

1. The resiliency link VLAN is handled internally in a very different way to other VLANs
2. Users should not be able to change the resiliency link VLAN's configuration, apart from the using resiliencylink commands.

There are two commands required to configure the resiliency VLAN:

```
stack resiliencylink
switchport resiliencylink
```

Once these commands are executed, the resiliency link is active.

Configuring the VLAN-based resiliency link

1. Once the switches are stacked via the stacking cables, you can create the resiliency VLAN and add ports to it:

   ```
   awplus#conf t
   ```

2. Enter configuration commands, one per line. End with Ctrl +Z.

   ```
   awplus(config)#stack resiliencylink vlan4001
   ```

3. Configure two ports on each member in the stack as the resiliency link ports:

```
awplus(config)#int port1.0.1
awplus(config-if)#switchport resiliencylink
awplus(config-if)#exit
awplus(config)#int port1.0.2
awplus(config-if)#switchport resiliencylink
awplus(config-if)#exit
awplus(config)#int port2.0.1
awplus(config-if)#switchport resiliencylink
awplus(config-if)#exit
awplus(config)#int port2.0.2
awplus(config-if)#switchport resiliencylink
awplus(config-if)#exit
awplus(config)#int port3.0.1
awplus(config-if)#switchport resiliencylink
awplus(config-if)#exit
awplus(config)#int port3.0.2
awplus(config-if)#switchport resiliencylink
awplus(config-if)#exit
awplus(config)#int port4.0.1
awplus(config-if)#switchport resiliencylink
awplus(config-if)#exit
awplus(config)#int port4.0.2
awplus(config-if)#switchport resiliencylink
awplus(config-if)#exit
```

4. Check that this has been configured correctly using the command:

```
awplus#show stack detail
Virtual Chassis Stacking detailed information

Stack Status:
-----------------------------------------------------------------
Normal operation
Operational Status              Enabled
Management VLAN ID                      4094
Management VLAN subnet address          192.168.255.0

Stack member 1:
-----------------------------------------------------------------
ID                                      1
Pending ID                              -
MAC address                             0015.77c2.4bb4
Last role change                        Wed Sep 16 10:38:17 2009
Product type                            x600-24Ts
Role                                    Backup Member
Priority                                128
Host name                               awplus-1
S/W version auto synchronization        On
Fallback config                         Not configured
Resiliency link                         Successful
Port 1.0.1 status                       Learnt neighbour 4
Port 1.0.2 status                       Learnt neighbour 2

Stack member 2:
-----------------------------------------------------------------
ID                                      2
Pending ID                              -
MAC address                             0015.7745.89d2
Last role change                        Wed Sep 16 10:38:16 2009
Product type                            x600-24Ts
Role                                    Backup Member
Priority                                128
Host name                               awplus
S/W version auto synchronization        On
Fallback config                         Not configured
Resiliency link                         Successful
Port 2.0.1 status                       Learnt neighbour 1
Port 2.0.2 status                       Learnt neighbour 3

Stack member 3:
-----------------------------------------------------------------
ID                                      3
Pending ID                              -
MAC address                             0015.77c2.4ba2
Last role change                        Wed Sep 16 10:38:16 2009
Product type                            x600-24Ts
Role                                    Active Master
Priority                                128
Host name                               awplus
S/W version auto synchronization        On
Fallback config                         Not configured
Resiliency link                         Configured
Port 3.0.1 status                       Learnt neighbour 2
Port 3.0.2 status                       Learnt neighbour 4

Stack member 4:
-----------------------------------------------------------------
ID                                      4
Pending ID                              -
MAC address                             0015.778e.62fa
Last role change                        Wed Sep 16 10:38:16 2009
Product type                            x600-24Ts
Role                                    Backup Member
Priority                                128
Host name                               awplus
S/W version auto synchronization        On
Fallback config                         Not configured
Resiliency link                         Successful
Port 4.0.1 status                       Learnt neighbour 2
Port 4.0.2 status                       Learnt neighbour 1
```

Note: There are no counters that can be viewed, because the resiliency link is only used

when a Backup Member loses connectivity with the Master via the stacking cables.

How VCS failover operates

When Backup Members lose Stack-XG connectivity with the Master, the resiliency-link determines whether the Master is still online. If no VCS healthcheck messages are received over the resiliency link within 2 seconds of failover, Backup Members assume that the Master is offline.

If the resiliency link is configured and active, but the interface is down, it is assumed that the Master is offline.

Failover situations in which the Backup Members know the master is rebooting always result in a Backup Member transitioning to Active Master. This occurs when the master is rebooted via the CLI, or when a node failover occurs due to processes on the master locking up or crashing.
If the Backup Member knows the Master is definitely online, then that Backup Member should become a Fallback Master or Disabled Master. These possible failover states are essentially the same as the Active Master (i.e. the master is running the active processes), but with differences in network configuration:

- Fallback Master
  The stack operates as usual, but is running an alternative configuration file called the fallback configuration (fallback-config) to avoid network conflicts with the master. This provides a back-up IP address for members that become isolated from the Master, although the fallback-config can also potentially contain the complete configuration for an alternative stack setup.
- Disabled Master
  The stack has disabled all its switch ports to avoid network conflicts, and is basically inactive. The stack is still assigned the Active process workload so the user can log in and reboot or reconfigure it. The separated slave's ports are taken down, which will stop network disruption as a result of LAG ports errantly being up. This is the default if the resiliency link is active but the fallback-config is not configured.

If the Backup Member has to leave the stack due to incompatible software, it should not cause network conflicts with the existing Master.

Health-check messages

Health-check messages are received if the Master is still online, but the stack will now split into two different 'stubs':

- The stub containing the existing Master continues operating as normal.
- The members of the other (Master-less) stub now use the fallback-config to form a second temporary stack. This utilizes the remaining stack members' resources without conflicting directly with the Master's configuration. If no fallback-config is specified for the stack, then the Master-less stub members disable their switch ports.

If no health-check messages are received, then the Master is assumed to be completely offline, and the other stack members can safely take over the Master's configuration.

The reboot rolling command

A major benefit of Virtual Chassis Stacking (VCS) is that it provides unit resiliency - even if one unit in a stack fails, the other stack members continue to forward data. It is highly desirable for this continuity of service to persist even when the stack is being rebooted. The purpose of the `reboot rolling` command is to reboot a stack in a manner that maintains continuity of service.

This command allows you to reboot a stack in a rolling sequence, so that at least one unit of the stack is active at any given time.

In this example, you have a stack of 3 x600 switches:

```
awplus#show stack
Virtual Chassis Stacking summary information

ID  Pending ID  MAC address        Priority  Status  Role
1   -           0015.77c2.4b7d     128       Ready   Backup Member
3   -           0015.77e8.a892     128       Ready   Backup Member
4   -           0015.77c9.73cb     128       Ready   Active Master

Operational Status                 Normal operation
Stack MAC address                  0015.77c9.73cb
```

Stack member 4 is the Active Master.

Use the command:

```
awplus#reboot rolling
```

The stack master reboots immediately with the configuration file settings. The remaining stack members reboot once the master has finished re-configuring.

```
Continue the rolling reboot of the stack? (y/n):y
awplus#22:11:07 awplus VCS[995]: Automatically rebooting stack member-
4 (MAC: 00 15.77c 9.73cb) due to Rolling reboot
URGENT: broadcast message:
System going down IMMEDIATELY!
        ... Rebooting at user request ...
```

During the reboot, another switch in the stack assumes the Active Master role. As soon as the original Active Master reloads, it becomes the Active Master again. Immediately after this, all of the other switches in the stack reboot simultaneously:

```
Active Master booting up:
Loading default configuration

done!
Received event network.configured

Rolling reboot, rebooting all other stack members, please wait for
stack to reform.
```

You can see in the Active Master's log that the other stack members (1 and 3) have rebooted:

```
2010 May 10 22:12:11 user.crit awplus-4 VCS[995]: Member 4 (0015.77c9.73cb) has become the Active
  Master
2010 May 10 22:12:37 local6.notice awplus VCS[995]: Link down event on stack link 4.0.2
2010 May 10 22:12:37 local6.notice awplus VCS[995]: Link down event on stack link 4.0.1
2010 May 10 22:13:32 local6.notice awplus VCS[995]: Link up event on stack link 4.0.1
2010 May 10 22:13:32 local6.notice awplus VCS[995]: Link down event on stack link 4.0.1
2010 May 10 22:13:32 local6.notice awplus VCS[995]: Link up event on stack link 4.0.2
2010 May 10 22:13:33 local6.notice awplus VCS[995]: Link down event on stack link 4.0.2
2010 May 10 22:13:36 local6.notice awplus VCS[995]: Link up event on stack link 4.0.1
2010 May 10 22:13:36 user.crit awplus VCS[995]: Member 3 (0015.77e8.a892) has joined stack
2010 May 10 22:13:36 user.notice awplus VCS[995]: Link between members 4 and 3 is up
2010 May 10 22:13:37 local6.notice awplus VCS[995]: Link up event on stack link 4.0.2
2010 May 10 22:13:37 user.crit awplus VCS[995]: Member 1 (0015.77c2.4b7d) has joined stack
2010 May 10 22:13:37 user.notice awplus VCS[995]: Link between members 4 and 1 is up
2010 May 10 22:13:37 user.notice awplus VCS[995]: Link between members 3 and 1 is up
```

Note: The `reload rolling` command is equivalent to the `reboot rolling` command.

The remote-login command

You can use the remote-login command on a stack master to log onto the CLI of a stack member.

Most of the time, once you are logged on to the stack member, entering commands gives the same results you would get if you were logged into the stack master. For example, the show ip interface command shows all IP interfaces configured on all switches in the stack - not just those on the stack member that you have connected to with the remote-login command. Configuration commands are still broadcast to all stack members.

There are however some show commands that execute locally. These include commands that display the switch's physical attributes, commands that access the file system, and commands related to feature licences.

1. **To login from the Stack master (stack member 1 in this case) to stack member 2:**

```
awplus#remote-login ?
  <1-8>  A specific stack member ID
awplus#remote-login 2
Type 'exit' to return to awplus.
AlliedWare Plus (TM) 5.3.4 05/04/10 11:59:17
awplus-2>en
awplus-2#
```

2. **Notice that the prompt has changed to reflect the stack member (2) that you are currently connected to. A directory listing now shows the files on stack member 2 only:**

```
awplus-2#dir *.cfg
      948 -rw- May  4 2010 20:59:48  flash:/default.cfg
      677 -rw- May  3 2010 18:39:04  flash:/zz.cfg
    2944 -rw- Mar 23 2010 12:55:40  flash:/ospfv3.cfg
```

3. You can delete a file from stack member 2 as if you are directly connected to it:

```
awplus-2#del zz.cfg
Delete flash:/zz.cfg? (y/n)[n]:y
Deleting..
Successful operation
awplus-2#
```

4. To return to the stack master, use the **exit** command:

```
awplus-2#exit
awplus#
```

The show license command

The show license command makes managing feature licenses on the stack members easy.

1. Connect to the stack member with the **remote-login** command:

```
awplus#remote-login 2
Type 'exit' to return to awplus.

AlliedWare Plus (TM) 5.3.4 05/04/10 11:59:17

awplus-2>en
awplus-2#
```

2. Use the **show license** command to view the current feature licenses on stack member 2:

```
awplus-2#show license
Software Feature Licenses
-------------------------------------------------------------------
Index                          : 0
License name                   : Base License
Customer name                  : Base License
Quantity of licenses           : 1
Type of license                : Full
License issue date             : 10-May-2010
License expiry date            : N/A
Features include               : VRRP OSPF-64 RADIUS-100 Virtual-MAC

Index                          : 1
License name                   : csg
Customer name                  : ATL-NZ (Internal Use Only)
Quantity of licenses           : 1
Type of license                : Full
License issue date             : 11-Aug-2009
License expiry date            : N/A
Features include              : BGP-64 PIM RIPNG VRRP OSPF-FULL VlanDT
OSPF-64
                            BGP-FULL IPv6Basic MLDSnoop BGP-5K RADIUS-100
                                   RADIUS-FULL PIM-100 ACCESS LAG-128
Virtual-MAC
```

3. To add a new license, paste in the license command generated by the AlliedWare Plus Licensing website:

```
awplus-2#license AT-FL-RAD-FULL
4pDI724ugtNcqlf8BmZMti2YEX6MSlS0GxDGCSlaf8aAYVDz
DtpZeg==
% Warning: license was only installed on member-2. Use the 'remote-
login' command to install it on all other stack members.
awplus-2#

awplus-2#show license
Software Feature Licenses
-----------------------------------------------------------------------
Index                          : 0
License name                   : Base License
Customer name                  : Base License
Quantity of licenses           : 1
Type of license                : Full
License issue date             : 10-May-2010
License expiry date            : N/A
Features include               : VRRP OSPF-64 RADIUS-100 Virtual-MAC

Index                          : 1
License name                   : csg
Customer name                  : ATL-NZ (Internal Use Only)
Quantity of licenses           : 1
Type of license                : Full
License issue date             : 11-Aug-2009
License expiry date            : N/A
Features include              : BGP-64 PIM RIPNG VRRP OSPF-FULL VlanDT
OSPF-64
                             BGP-FULL IPv6Basic MLDSnoop BGP-5K RADIUS-100
                            RADIUS-FULL PIM-100 ACCESS LAG-128 Virtual-MAC

Index                          : 2
License name                   : AT-FL-RAD-FULL
Customer name                  : ATL-NZ L3 CSG
Quantity of licenses           : 1
Type of license                : Full
License issue date             : 09-May-2010
License expiry date            : N/A
Features include               : RADIUS-FULL
```

Provisioning

Provisioning allows you to pre configure ports that are not yet physically present in a switch, and units not yet physically present in a stack. If a switch allows hot-swappable XEMs, then provisioning allows the ports of these yet-to-be-inserted XEMs to be preconfigured prior to the XEMs' insertion. Similarly, if you know that a switch will be added to a stack, you can pre configure that new switch in preparation for its addition to the stack.

You can either pre-configure ports or switches that have not yet been installed, or you can load a configuration that references these ports. Provisioning also automatically keeps track of the configuration that was present on XEMs that have been hotswapped out of a switch, or on units that have been removed from a stack. Provisioning keeps a placeholder for a XEM or switch which has been hotswapped out.

If you provision a switch or bay, then decide later to change the stack member ID or bay number before it has been installed, you must unprovision (no switch <stack ID> bay/switch) the switch or bay first.

Provisioning a bay

With the **show sys** command, you can see that the stack member 2 x900-24XT switch does not have a XEM in bay 2:

```
awplus#show sys
Stack System Status                                Wed May 05 00:04:16 2010

Stack member 1:

Board      ID  Bay   Board Name                       Rev   Serial number
-------------------------------------------------------------------
Base       271        x900-24XS                        B-0   P1HF7801H
Expansion  272  Bay1  XEM-1XP                          B-0   41AR67008
Expansion  285  Bay2  XEM-STK                          A-0   M1L18400R
PSU        212  PSU1  AT-PWR01-AC                       F-1   73173269
Fan module 214  PSU2  AT-FAN01                         F-1   73169578
-------------------------------------------------------------------
RAM:  Total: 513372 kB Free: 396680 kB
Flash: 31.0MB Used: 15.9MB Available: 15.1MB
-------------------------------------------------------------------
Environment Status : Normal
Uptime             : 0 days 00:55:48
Bootloader version : 1.0.9


Stack member 2:

Board      ID  Bay   Board Name                       Rev   Serial number
-------------------------------------------------------------------
Base       270        x900-24XT                        A-0   M1QH78003
Expansion  285  Bay1  XEM-STK                          A-0   M1L17400G
PSU        212  PSU2  AT-PWR01-AC                       B-1   61410709
-------------------------------------------------------------------
RAM:  Total: 513372 kB Free: 410648 kB
Flash: 63.0MB Used: 30.9MB Available: 32.1MB
-------------------------------------------------------------------
Environment Status : Normal
Uptime             : 0 days 00:25:34
Bootloader version : 1.0.9
```

You can see that Stack member 1 is the Master, and that you are connected to the console port on this switch:

```
awplus#show stack
Virtual Chassis Stacking summary information

ID  Pending ID  MAC address        Priority  Status  Role
1   -           0000.cd27.c4bf     128       Ready   Active Master
2   -           0000.cd28.0801     128       Ready   Backup Member

Operational Status                 Normal operation
Stack MAC address                  0000.cd27.c4bf
```

On the Stack Master (stack member 1) you can provision a XEM-12 for Stack member 2 in bay 2 (which is currently empty):

```
awplus(config)#switch 2 bay 2 provision xem-12

switch 1 provision x900-24
switch 1 bay 1 provision xem-1
switch 2 provision x900-24
switch 2 bay 2 provision xem-12
!
interface port2.0.1-2.0.24
 switchport
 switchport mode access
!
interface port2.2.1-2.2.12
 switchport
 switchport mode access
!
```

Note: Note that the switch automatically provisions all currently installed switches and XEMs as it boots up. It doesn't provision the actual stacking XEMs.

You can see above that you now have ports 2.2.1-2.2.12 available for configuration in the running-config, even though stack member 2 does not yet actually have a 12 port XEM (XEM-12) physically installed in bay 2.

This means that you can now configure these ports ready for when the XEM-12 is installed:

```
awplus(config)#int port2.2.1
awplus(config-if)#switchport access vlan 2
```

Commands can refer to ports on that provisioned XEM as though it were already present. Once a XEM is hotswapped into bay 2, the "switch 2 bay 2 provision xem-12" still shows in the running configuration, along with the other installed switches and XEMs. If you remove the XEM, the provisioning for it remains along with the configuration for its associated ports.

What happens when a provisioned XEM is hotswapped out?

In the example below, stack member 1 has a XEM-1XP installed in bay 1 and its port (port1.1.1) is configured as a trunk.

```
switch 1 provision x900-24
switch 1 bay 1 provision xem-1
switch 2 provision x900-24
!
interface port1.1.1
 switchport
 switchport mode trunk
 switchport trunk allowed vlan all
 switchport trunk native vlan none
!
```

If the XEM-1XP is hotswapped out of bay 1:

```
awplus#08:23:05 awplus HPI: HOTSWAP Pluggable 1.1.1 hotswapped out:
FTRX-1411-3
08:23:05 awplus HPI: HOTSWAP XEM 1 hotswapped out: XEM-1XP
08:23:05 awplus EXFX[1268]: Handle event: bay 1 hsState 4 bt 272 br 0
08:23:05 awplus NSM[1121]: Removal event on bay 1.1 has been completed
```

You can see that the configuration associated with this port is still in the running configuration:

```
interface port1.1.1
 switchport
 switchport mode trunk
 switchport trunk allowed vlan all
 switchport trunk native vlan none
!
```

What happens when the XEM is hotswapped back in?

If the XEM-1xp is hotswapped back into bay 1:

```
awplus#08:25:18 awplus HPI: HOTSWAP XEM 1 hotswapped in: XEM-1XP
08:25:18 awplus HPI: HOTSWAP Pluggable 1.1.1 hotswapped in: FTRX-1411-3
08:25:18 awplus EXFX[1268]: Handle event: bay 1 hsState 2 bt 272 br 1
08:25:22 awplus EXFX[1268]: Board XEM-1XP inserted into bay 1
08:25:22 awplus EXFX[1268]: Please wait until configuration update is completed
08:25:22 awplus IMI[1123]: All users returned to config mode while switch synch
ronization is in progress.
08:25:22 awplus VCS[1118]: XEM-1XP has been inserted into bay 1.1
08:25:22 awplus NSM[1121]: Insertion event on bay 1.1 has been completed
08:25:23 awplus IMI[1123]: Configuration update completed for port1.1.1
```

You can see above that port1.1.1 has had its configuration updated from the running config.

What happens if a different type of XEM is hotswapped in?

If the XEM-1XP is hotswapped out and a different type of XEM (in this case a XEM-12T) is hotswapped into bay 1 instead:

```
awplus#08:28:48 awplus HPI: HOTSWAP Pluggable 1.1.1 hotswapped out:
FTRX-1411-3
08:28:48 awplus HPI: HOTSWAP XEM 1 hotswapped out: XEM-1XP
08:28:48 awplus EXFX[1268]: Handle event: bay 1 hsState 4 bt 272 br 0
08:28:48 awplus NSM[1121]: Removal event on bay 1.1 has been completed

awplus#08:29:05 awplus HPI: HOTSWAP XEM 1 hotswapped in: XEM-12T
08:29:05 awplus EXFX[1268]: Handle event: bay 1 hsState 2 bt 274 br 2
08:29:08 awplus EXFX[1268]: Board XEM-12T inserted into bay 1
08:29:08 awplus EXFX[1268]: Please wait until configuration update is
completed
08:29:08 awplus IMI[1123]: All users returned to config mode while
switch synch
ronization is in progress.
08:29:08 awplus VCS[1118]: XEM-12T has been inserted into bay 1.1
08:29:09 awplus NSM[1121]: Insertion event on bay 1.1 has been
completed
08:29:11 awplus IMI[1123]: Configuration update completed for
port1.1.1-1.1.12
```

You can see that the provisioning has been modified to reflect the actual hardware installed:

```
switch 1 provision x900-24
switch 1 bay 1 provision xem-12
switch 2 provision x900-24
!
interface port1.1.1-1.1.12
 switchport
 switchport mode access
!
```

The running configuration now has ports1.1.1-1.1.12, which are the 12 ports belonging to the XEM-12T in bay.

Provisioning a switch

This example involves the future addition of a switch to a standalone switch to form a stack:

```
awplus#sh sys
Switch System Status                              Wed May 05 14:34:12 2010

Board      ID  Bay   Board Name                        Rev   Serial number
-------------------------------------------------------------------------
---
Base       287       x900-12XT/S                        A-0   M1NB7C023
Expansion  285  Bay1  XEM-STK                            A-0   A1L18305D
-------------------------------------------------------------------------
---
RAM:  Total: 513372 kB Free: 422964 kB
Flash: 63.0MB Used: 46.0MB Available: 17.0MB
```

The current switch has an ID (stack member) of 2:

```
awplus#show stack
Virtual Chassis Stacking summary information

ID  Pending ID  MAC address        Priority  Status  Role
2   -           0000.cd28.bff7     128       Ready   Active Master

Operational Status                 Standalone unit
Stack MAC address                  0000.cd28.bff7
```

Procedure

1. Provision stack member 1 so that you can configure the future stack member's ports before you actually have the second switch connected:

```
awplus(config)#switch 1 provision ?
  x600-24   Provision an x600-24 switch
  x600-48   Provision an x600-48 switch
  x900-12   Provision an x900-12 switch
  x900-24   Provision an x900-24 switch
  x908      Provision an x908 switch
```

2. Select the switch model to be connected in the future. You can only stack, and therefore provision, switches of the same basic model. For example, if you try to provision an x900-24 switch for stack member 1, and the existing switch (stack member 2) is an x900-12, you get the following error message.

```
awplus(config)#switch 1 provision x900-24
% Board class x900-24 is incompatible with existing stack members.
```

3. You can successfully provision an x900-12 as follows:

```
awplus(config)#switch 1 provision x900-12
```

The running-config shows that you can now configure the ports (1.0.1-1.0.12) on provisioned stack member 1:

```
switch 1 provision x900-12
switch 2 provision x900-12
!
interface port1.0.1-1.0.12
 switchport
 switchport mode access
!
```

Note: The configuration applied to ports1.0.1-1.012 is the default port configuration. The port trunk configuration provisioned for the XEM-1XP is completely discarded when the XEM-12S is hotswapped in instead.

Reprovisioning

To change the provisioning, for example if you wanted to change a provisioned x600-24 to an x600-48, you must first execute no switch x provision followed by switch x provision x600-48, as switch x provision fails if there is existing provisioning. However, this process means you will lose all the configuration for portx.0.1-24.

Using switch x reprovision x600-48 lets you change the provisioning without losing any existing configuration (within the limits of the respective port counts of the two device types). It allows you to change existing provisioning - provided no actual hardware is present.

You can also reprovision a XEM in a bay. The below example provisions a XEM-12 in bay 2 on switch member 2:

```
awplus(config)#switch 2 bay 2 provision xem-12
```

You can then configure port2.2.1 (the first port on the XEM-12) as follows:

```
awplus(config)#int port2.2.1
awplus(config-if)#swi access vlan 2
```

If you decide to use a XEM-1XP instead of the XEM-12, you can reprovision this change and keep the configuration for any ports that overlap - in this case only port2.1.1:

```
awplus(config)#switch 2 bay 2 reprovision xem-1
```

If you instead remove the provisioned XEM and added another, the overlapping port (port2.2.1) is deleted and any configuration on it lost:

```
awplus(config)#no switch 2 bay 2 provision
awplus(config)#switch 2 bay 2 provision xem-1
```

Security

Web Auth proxy

There are two scenarios in which this feature can be used:

When you manually specify the supplicant's web proxy port

1. The first is when standard AlliedWare Plus Web Authentication intercepts the supplicant's initial TCP port 80 connection to a web page and sends it the Web Authentication Login page. If the supplicant is configured to use a web proxy, then it will usually be using TCP port 8080 (or another user configured port number). In this case Web Authentication cannot intercept the connection.

To overcome this limitation use the command **auth-web-server intercept-port** to tell the switch which TCP port it should intercept, and then send the Web Authentication Login page to the supplicant.

This is configured by the following command:

```
Authenticator(co)#auth-web-server intercept-port <port-number>
```

The switch will still intercept a connection to a standard web page on TCP port 80 as well – this command adds an additional port.

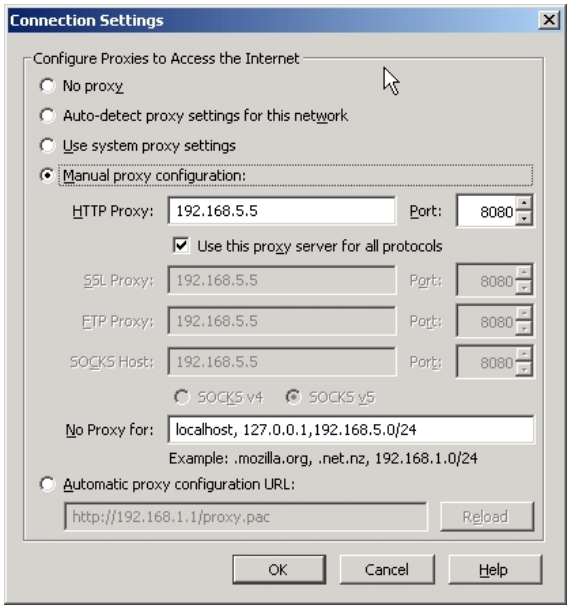
The output from the command **show auth-web-server** gives us all of the information on the web authentication configuration, including the HTTP Intercept port that we have configured:

```
Authenticator#show auth-web-server
Web authentication server
  Server status: enabled
  Server mode: intercept
  Server address: 10.33.24.28/0
  HTTP Port No: 80
  Security: disabled
  Certification: default
  SSL Port No: 443
  HTTP Intercept Port No: 8080
  Redirect URL: --
  Redirect Delay time: default
  HTTP Redirect: enabled
  Session keep: disabled
  Blocking mode: disabled (cur session: 0)
  PingPolling: disabled
  PingInterval: 30
  Timeout: 1
  FailCount: 5
  ReauthTimerRefresh: disabled
```

One important point in using the **auth-web-server intercept-port** command, in conjunction with a proxy server configured in the web browser, is that the network on which the proxy server is located on must be added as a 'No Proxy' network.

For example, consider the case where the web browser is configured to use 192.168.5.5 TCP port 8080 as its web proxy.

In this case we must also add the network 192.168.5.0/24 into the 'No Proxy for:' list, as shown below:



If the network on which the proxy server is located on is not added to the **No Proxy for**: list, a 'proxy packet loop' will occur. The 'proxy packet loop' occurs because the switch intercepts the HTTP traffic from the supplicant's browser to the web proxy, and sends it a 'fake' response redirecting it to the Web Authentication Login page – in this case the switch at 192.168.5.1.

However, the web browser is configured to use the web proxy at 192.168.5.5 for traffic to network 192.168.5.0/24, and so sends a new request to 192.168.5.5 instead. The switch will intercept this request and the process begins again.

Properly configured, when the supplicant opens their web browser, the switch will send it the Web Authentication Login page. Once it has been successfully authenticated the supplicant will use its configured web proxy for any external web pages.

```
Interface port1.0.9
  authenticationMethod: web
  totalSupplicantNum: 1
  authorizedSupplicantNum: 1
    macBasedAuthenticationSupplicantNum: 0
    dot1xAuthenticationSupplicantNum: 0
    webBasedAuthenticationSupplicantNum: 1
    otherAuthenticationSupplicantNum: 0

Interface   VID  Mode MAC Address    Status           IP Address
Username
=========== ==== ==== ============== ================ ===============
port1.0.9   5    W    0008.0d5e.c216 Authenticated    192.168.1.200   web
```

When the supplicant uses Web Proxy Auto-Discovery

2. The second scenario in which the Web Auth proxy feature is used is if the supplicant is configured to use WPAD (Web Proxy Auto-Discovery). The supplicant's web browser will use TCP port 80 as usual, and so can be intercepted by Web Authentication as normal, and the Web Authentication Login page sent. However, after authentication, the supplicant does not know where to get the WPAD file (usually named proxy.pac) which tells it what its web proxy is, so the supplicant cannot access external web pages. We can use the **auth-web-server dhcp wpad-option** to tell the supplicant where it can get this file from. The switch itself can be specified as the source for this file, and it can deliver it to the supplicant on request.

When a supplicant that is configured to use WPAD opens their web browser, before fetching its first web page, it sends the local DHCP server a DHCP INFORM query, and uses the URL from the WPAD option in the server's reply to determine where it should get its WPAD configuration file.

This DHCP Inform query is only sent when the web browser is first opened, so it will not request this again after authentication unless the web browser is closed and opened again. So, when using web authentication with WPAD, it needs the switch to reply to this DHCP Inform query with the URL from which the supplicant can get the proxy.pac file. If it does not get a response then, after the supplicant has been authenticated, it will not have its web proxy information and will not be able to access external web pages.

To configure the switch for web authentication where the supplicant is using WPAD, you can use the **auth-web-server dhcp wpad-option** to tell the supplicant where it should get the proxy.pac file from.

Normally this will be the authenticating switch itself. The proxy.pac file contains the URL and/or IP address of the web proxy server that the supplicant should use.

In the example below we are using the **auth-web-server dhcp** feature to tell the supplicant where it should get its WPAD proxy.pac file from – 192.168.1.1 (the authenticating switch).

As well as supplying the WPAD information, it will also supply the supplicant with its IP address information via DHCP:

```
auth-web-server dhcp ipaddress 192.168.1.1/24
auth-web-server dhcp wpad-option 192.168.1.1/proxy.pac
```

The proxy.pac file can be copied onto the switch via the following method:

```
Authenticator#copy tftp://192.168.1.100/proxy.pac proxy-autoconfig-
file
Copying...
Successful operation
```

An example of a proxy.pac file can be seen below – the web proxy here is set to 192.168.5.5:

```
Authenticator#show proxy-autoconfig-file
function FindProxyForURL(url, host)
{

if (isInNet(myIpAddress(), "192.168.5.5", "255.255.255.0")

return "PROXY [Proxy Address]:[Port]";

else

return "DIRECT";

}
```

In the packet capture below, we can see that the supplicant now knows that it should get its proxy.pac file from 192.168.1.1 (the authenticating switch) and has requested a copy of it:

```
10  2013-08-13 192.168.5.19192.168.1.1 HTTP 434 GET /proxy.pac HTTP/1.1
```

And the switch has replied with a copy of the file:

```
11   2013-08-13   192.168.1.1192.168.5.19HTTP231HTTP/1.1 200 OK  (text/
plain)

Hypertext Transfer Protocol
    HTTP/1.1 200 OK\r\n
        [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
            [Message: HTTP/1.1 200 OK\r\n]
            [Severity level: Chat]
            [Group: Sequence]
        Request Version: HTTP/1.1
        Status Code: 200
        Response Phrase: OK
    Date: Tue, 13 Aug 2013 01:04:39 GMT\r\n
    Last-Modified: Mon, 12 Aug 2013 22:40:01 GMT\r\n
    Etag: "52096441.b1"\r\n
    Content-Type: text/plain\r\n
    Content-Length: 177\r\n
        [Content length: 177]
    Connection: close\r\n
    \r\n
Line-based text data: text/plain
    function FindProxyForURL(url, host)\r\n
    \r\n
    {\r\n
    \r\n
    if (isInNet(myIpAddress(), "192.168.5.5", "255.255.255.0")\r\n
    \r\n
    return "PROXY [Proxy Address]:[Port]";\r\n
    \r\n
    else\r\n
    \r\n
    return "DIRECT";\r\n
    \r\n
    }
```

So, now the supplicant's web browser knows that it should use IP address 192.168.5.5 as its web proxy and can access external web pages.

Two-step authentication

Single authentication methods (either user or device authentication) have a potential security risk in that an unauthorised user can access the network with an authorised device, and an authorised user can access the network with an unauthorised device.

Two-Step Authentication can authenticate both the device as well as the user if both of these steps are successful, does the supplicant becomes authenticated. If the first authentication step fails, then the second step is not started.

The following authentication sequences are supported for Two-Step Authentication.

- MAC Authentication followed by 802.1X Authentication
- MAC Authentication followed by Web Authentication
- 802.1X Authentication followed by Web Authentication

Combinations of Two-Step Authentication and AuthFail VLAN / Guest VLAN / Dynamic VLAN on the same interface are supported:

Two-step authentication and AuthFail VLAN

If a supplicant fails either the first or second step, it is assigned to the AuthFail VLAN, if configured.

Two-step authentication and guest VLAN

If a supplicant fails either the first or second step, it is assigned to the Guest VLAN, if configured.

Two-step authentication and dynamic VLAN

If a supplicant is successfully authenticated by both the first and second steps, it is assigned to the Dynamic VLAN if **auth dynamic-vlan-creation** is configured on the port. The VLAN assignment is in the RADIUS-Accept packet of the second authentication step. If a VLAN assignment is configured for the first authentication step in the RADIUS Server, this is ignored. The supplicant will only be dynamically assigned to a VLAN by the Authenticating switch once both authentication steps are successful.

Examples

MAC authentication followed by 802.1X authentication

Port configuration

```
interface port1.0.6
switchport mode access
 auth-mac enable
 dot1x port-control auto
 auth dynamic-vlan-creation
 auth two-step enable
```

The supplicant's device is automatically MAC authenticated then, if that is successful, the supplicant must then supply their username and password for dot1x authentication.

After authentication, the command **show auth-mac supplicant brief** displays:

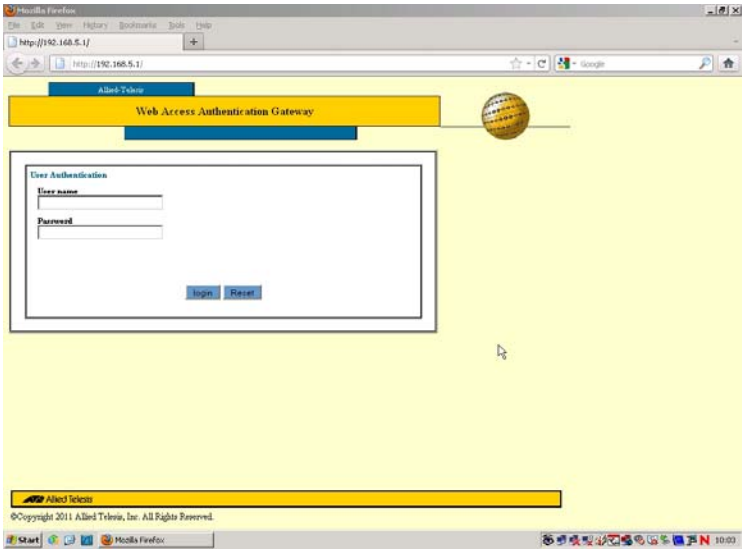
```
Authenticator#show auth-mac supplicant brief

Interface port1.0.6
  authenticationMethod: dot1x/mac
  Two-Step Authentication
    firstMethod: mac
    secondMethod: dot1x
  totalSupplicantNum: 1
  authorizedSupplicantNum: 1
    macBasedAuthenticationSupplicantNum: 0
    dot1xAuthenticationSupplicantNum: 1
    webBasedAuthenticationSupplicantNum: 0
    otherAuthenticationSupplicantNum: 0

Interface   VID  Mode MAC Address    Status       IP Address     Username
=========== ==== ==== ============== ================ ===============
port1.0.6   5    D    0008.0d5e.c216 Authenticated    --           dot1x
```

MAC authentication followed by web authentication

Port configuration

```
interface port1.0.7
switchport mode access
 auth-mac enable
 auth-web enable
 auth dynamic-vlan-creation
 auth two-step enable
```

The supplicant's device is automatically MAC authenticated then, if that is successful, the supplicant receives the Web Authentication Login page when their web browser is opened.



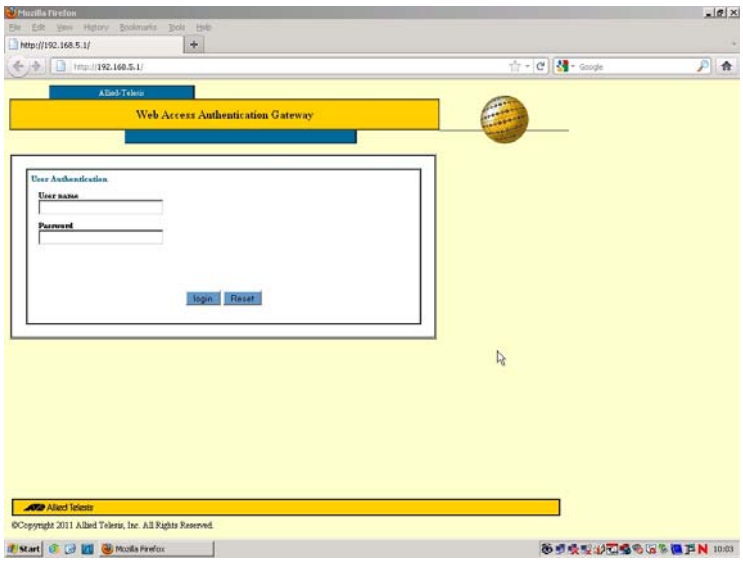
After successful authentication, the command **show auth-mac supplicant brief** displays:

```
Authenticator#show auth-mac supplicant brief

Interface port1.0.7
  authenticationMethod: mac/web
  Two-Step Authentication
    firstMethod: mac
    secondMethod: web
  totalSupplicantNum: 1
  authorizedSupplicantNum: 1
    macBasedAuthenticationSupplicantNum: 0
    dot1xAuthenticationSupplicantNum: 0
    webBasedAuthenticationSupplicantNum: 1
    otherAuthenticationSupplicantNum: 0

Interface   VID  Mode MAC Address    Status       IP Address     Username
=========== ==== ==== ============== ================ ===============
port1.0.7   5    W    0008.0d5e.c216 Authenticated    192.168.1.200   web
```

802.1X authentication followed by web authentication

```
interface port1.0.8
switchport mode access
 auth-web enable
 dot1x port-control auto
 auth dynamic-vlan-creation
 auth two-step enable
```

Here the supplicant must first enter their username and password for dot1x authentication then, if that is successful, the supplicant receives the Web Authentication Login page when their web browser is opened.



After successful authentication, the command **show dot1x supplicant brief** displays:

```
Authenticator#show dot1x supplicant brief

Interface port1.0.8
  authenticationMethod: dot1x/web
  Two-Step Authentication
    firstMethod: dot1x
    secondMethod: web
  totalSupplicantNum: 1
  authorizedSupplicantNum: 1
    macBasedAuthenticationSupplicantNum: 0
    dot1xAuthenticationSupplicantNum: 0
    webBasedAuthenticationSupplicantNum: 1
    otherAuthenticationSupplicantNum: 0

Interface   VID  Mode MAC Address    Status      IP Address     Username
=========== ==== ==== ============== ================ ===============
port1.0.8   5    W    0008.0d5e.c216 Authenticated    192.168.1.200   web
```

Forwarding DNS packets using Auth-web forward command

In auth-web-server mode '*intercept*' and '*promiscuous*', the AlliedWare Plus switch will capture ARP, DNS, and HTTP packets from the supplicant, so that it can send the supplicant the Web Authentication Login page.

Initially, in auth-web-server mode '*none*', the AlliedWare Plus switch will only capture HTTP packets from the supplicant and will not pass through other types of unicast traffic.

All broadcast and multicast packets are passed through the AlliedWare Plus switch in all modes.

Before the supplicant can send an HTTP request, and have it intercepted by the switch in auth-web-server mode '*none*', it must use DNS to resolve the URL of the initial web page that the web browser is attempting to get to.

In this case we can use the **auth-web forward** command to tell the switch to send any DNS packets from the supplicant to a DNS Server's IP address:

```
interface port1.0.9
switchport mode access
 auth-web enable
 auth dynamic-vlan-creation
 auth-web forward 192.168.1.10 dns
```

Here the switch will forward any DNS packets, received on this port, to a DNS Server at 192.168.1.10.

Once the supplicant has been able to resolve the initial web page URL, then it will send an HTTP request for this page. This HTTP request will be intercepted by the switch and the Web Authentication Login page will be sent to the supplicant.

Because the supplicant has not yet been authenticated, we can only use the **auth-web forward** command to send packets to a device on the same VLAN as that on which the unauthorised supplicant is on – in this case VLAN1.

The options for the auth-web forward command are:

```
A.B.C.D  Destination IPv4 address (default: any)
  arp      ARP packets
  dhcp     DHCP packets (67/udp)
  dns      DNS packets (53/udp)
  tcp      TCP protocol
  udp      UDP protocol
```

We can configure auth-web forwarding for ARP and DHCP if required, but this is not usually needed as these protocols normally use broadcasts, which will be passed by the AlliedWare Plus switch.

Configuring port-security, but not configuring a port-security maximum

If port-security is configured on an interface, but the **port-security maximum** is not explicitly configured, then this can cause the CPU to show higher than normal utilisation. The **port-security maximum** command specifies how many MAC addresses can be learned on a port.

The reason for the increase in CPU utilisation is because the default port-security maximum is 0, which means that the switch will be continually attempting to learn MAC addresses on the port, but will then have to discard them.

```
interface port2.0.1
 switchport
 switchport mode access
 switchport port-security
 switchport port-security maximum <0-256>
```

To avoid this higher than normal CPU utilization, make sure you explicitly set the port-security maximum to 1 or higher.

Web Authentication enhancements

From maintenance release **5.4.3-2.5**, there are several small web authentication enencements which are discussed below:

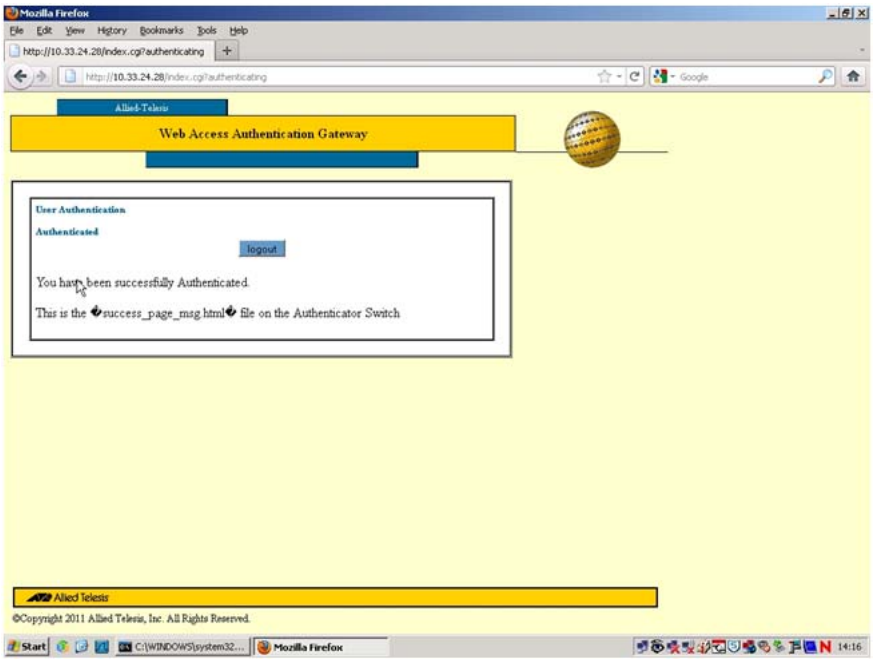
User customised message

It is now possible to configure a message which is displayed when the supplicant has been successfully authenticated. There is no additional configuration needed to do this, but the files must be named **success_page_msg.html** or **message.html**.

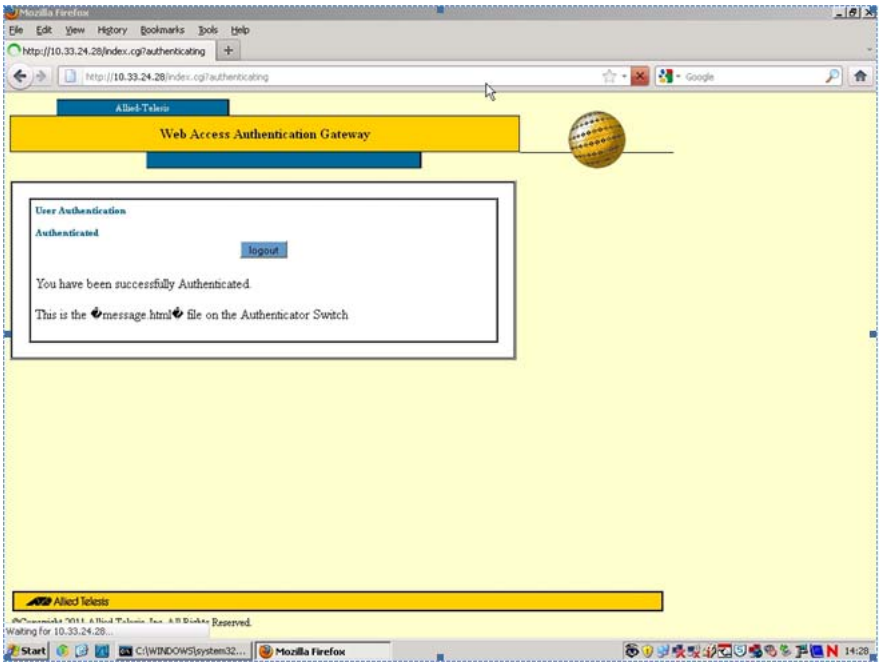
If the switch has both files in flash, it will use the **success_page_msg.html** file.

```
Authenticator#dir *.html
    21959 -rw- Jul 24 2013 02:08:14  flash:/
   success_page_msg.html
    21921 -rw- Jul 24 2013 02:07:29  flash:/message.html
```

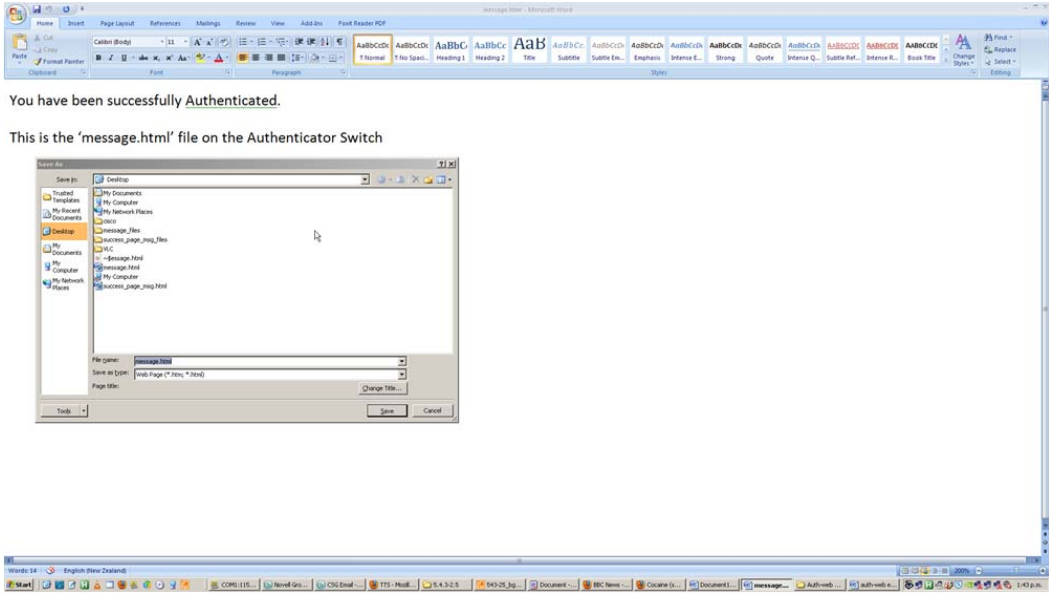
Here we see the **success_page_msg.html** file, which we created, displayed after a successful login.The user configured message is only displayed in the white **User Authentication** box on the **Web Access Authentication Gateway** page from the AlliedWare Plus switch. The rest of the web page is not affected, and cannot be configured, by the user configured message:



Here the success_page_msg.html file does not exist on the switch, so the message.html file we created is displayed instead:



The html files used above were just simple text files created in Microsoft Word, then saved as an .html file:



Configurable redirect-url delay time

The existing 'redirect-url' auth-web feature allows the supplicant's web browser to be redirected to a user configured web page after the auth-web authentication is successful. In some cases a delay between the success message and the actual redirection may be required.

The amount of time that elapses between a successful login, and the redirection to a configured URL, can now be set to a period between `5-60 seconds`. The authentication 'Success' page from the AlliedWare Plus switch is displayed to the user during the delay period.

`Example:`

```
Authenticator(config)#auth-web-server redirect-url http://www.google.com
Authenticator(config)#auth-web-server redirect-delay-time 30
```

Gateway registration function

This feature is useful in the situation where the supplicant gets its IP information (IP address, subnet mask, default gateway, DNS, etc.) via an external DHCP Server, which gives it a long lease time (as opposed to the AW+ auth-web-server DHCP server, which gives very short lease times of 20-60 seconds).

The sequence of events is as follows:

- The supplicant opens their web browser and attempts to get to a web page by sending an HTTP Get Request packet to an external Web host. The switch's Web Authentication must hijack that packet, so that it can authenticate the supplicant first.

  To achieve this, when the supplicant sends the ARP Request for the MAC address of its default gateway, the AW+ switch replies to this ARP Request with its own MAC address instead.

- The AW+ switch then sends the supplicant the auth-web page, and the supplicant enters the username/password and is authenticated.

The problem that can occur with a long DHCP lease is that the supplicant should be able to connect to the network now that it has been authorised but, as the AW+ authenticator faked the supplicant's default gateway IP address with the AW+ Authenticator MAC address, the supplicant cannot communicate directly to the actual gateway.

To get around this problem, the `auth-web-server gateway` command has been introduced.

The MAC address of the original supplicant's gateway is looked up in the AW+ Authenticator switch's ARP table from the gateway IP address configured by the new CLI command:

```
auth-web-server gateway 192.168.1.1 vlan 5
```

(the IP address of the supplicant's gateway and which VLAN it is on)

- If the ARP entry does not already exist, the switch resolves the ARP by sending an ARP Request for the gateway's MAC address.
- The AW+ switch then sends a Gratuitous ARP that tells the supplicant the correct MAC address of its' default gateway. The supplicant will update its' ARP entry, and can then connect to the network.

Note: This feature is only applicable in cases where the supplicant is in the same VLAN before and after authentication. If the supplicant is assigned to another VLAN after authentication, the client will not have access to the network until its DHCP lease expires, and it receives its new IP information for the new VLAN from the DHCP Server.

This feature works with auth-web server configured for either promiscuous or intercept mode.

Diagnostics

CPU usage spikes

Note: This issue occurred in AlliedWare Plus Release 5.2.2, and was resolved in AlliedWare Plus Release 5.3.1.

The issue

The CPU usage has the potential to spike to 40% every 15 seconds, as shown below:

```
Stack member 1:

Per second CPU load history

100
90
80
70
60
50
40       *              *              *              *
30
20
10 ****** ************** ************** ************** ********
|....|....|....|....|....|....|....|....|....|....|....|....
Oldest Newest
CPU load% per second (last 60 seconds)
* = average CPU load%
```

These spikes in CPU usage are caused by the SNMP protocol. However, this occurs even when the SNMP process is disabled on the switch:

```
Dong_Bu_Ring#sh snmp-server
SNMP enable ........ No
SNMPv3 engine ID (configured) ....... Not set
SNMPv3 engine ID (actual)............ Not set
```

If you turn on Terminal Monitor, you will see that the following SNMP log messages occur every 15 seconds. This shows that SNMP is still polling the software protocol modules, even though it is disabled:

```
15:19:29 Dong_Bu_Ring LACP[1966]: AgentX: ping, Operational state, fail 0
15:19:29 Dong_Bu_Ring LACP[1966]: AgentX: pinging:
15:19:29 Dong_Bu_Ring LACP[1966]: AgentX: build Ping-PDU
15:19:29 Dong_Bu_Ring LACP[1966]: ->  AgentX Header:
15:19:29 Dong_Bu_Ring LACP[1966]:       Version: 1
15:19:29 Dong_Bu_Ring LACP[1966]:       Type: 13 (Ping)
15:19:29 Dong_Bu_Ring LACP[1966]:       Flags: 00
15:19:29 Dong_Bu_Ring LACP[1966]:       <reserved>: 0
15:19:29 Dong_Bu_Ring LACP[1966]:       Session ID: 13 (0x0D)
15:19:29 Dong_Bu_Ring LACP[1966]: ->      Integer: 13 (0x0D)
15:19:29 Dong_Bu_Ring IMI[1928]: AgentX: ping, Operational state, fail 0
15:19:29 Dong_Bu_Ring IMI[1928]: AgentX: pinging:
15:19:29 Dong_Bu_Ring IMI[1928]: AgentX: build Ping-PDU
15:19:29 Dong_Bu_Ring IMI[1928]: ->  AgentX Header:
15:19:29 Dong_Bu_Ring IMI[1928]:       Version: 1
15:19:29 Dong_Bu_Ring IMI[1928]:       Type: 13 (Ping)
15:19:29 Dong_Bu_Ring IMI[1928]:       Flags: 00
15:19:29 Dong_Bu_Ring IMI[1928]:       <reserved>: 0
15:19:29 Dong_Bu_Ring IMI[1928]:       Session ID: 14 (0x0E)
15:19:29 Dong_Bu_Ring IMI[1928]: ->      Integer: 14 (0x0E)
15:19:29 Dong_Bu_Ring LACP[1966]:       Transaction ID: 0 (0x00)
15:19:29 Dong_Bu_Ring LACP[1966]: ->      Integer: 0 (0x00)
15:19:29 Dong_Bu_Ring LACP[1966]:       Packet ID: 621120 (0x97A40)
15:19:29 Dong_Bu_Ring LACP[1966]: ->      Integer: 621120 (0x97A40)
15:19:29 Dong_Bu_Ring LACP[1966]:       Dummy Length: -(
15:19:29 Dong_Bu_Ring LACP[1966]: ->      Integer: 0 (0x00)
15:19:29 Dong_Bu_Ring LACP[1966]:     Payload
15:19:29 Dong_Bu_Ring LACP[1966]: ->      Integer (length of PDU) : 0 (0x00)
15:19:29 Dong_Bu_Ring LACP[1966]: AgentX: built packet okay
15:19:29 Dong_Bu_Ring LACP[1966]: AgentX: sending PDU-XDUMP:
15:19:29 Dong_Bu_Ring NSM[2005]: AgentX: ping, Operational state, fail 0
15:19:29 Dong_Bu_Ring NSM[2005]: AgentX: pinging:
15:19:29 Dong_Bu_Ring NSM[2005]: AgentX: build Ping-PDU
15:19:29 Dong_Bu_Ring NSM[2005]: ->  AgentX Header:
15:19:29 Dong_Bu_Ring NSM[2005]:       Version: 1
15:19:29 Dong_Bu_Ring NSM[2005]:       Type: 13 (Ping)
15:19:29 Dong_Bu_Ring NSM[2005]:       Flags: 00
15:19:29 Dong_Bu_Ring NSM[2005]:       <reserved>: 0
15:19:29 Dong_Bu_Ring NSM[2005]:       Session ID: 12 (0x0C)
15:19:29 Dong_Bu_Ring NSM[2005]: ->      Integer: 12 (0x0C)
15:19:29 Dong_Bu_Ring 802.1X[1809]: AgentX: ping, Operational state, fail 0
15:19:29 Dong_Bu_Ring 802.1X[1809]: AgentX: pinging:
15:19:29 Dong_Bu_Ring 802.1X[1809]: AgentX: build Ping-PDU
15:19:29 Dong_Bu_Ring 802.1X[1809]: ->  AgentX Header:
15:19:29 Dong_Bu_Ring 802.1X[1809]:       Version: 1
15:19:29 Dong_Bu_Ring 802.1X[1809]:       Type: 13 (Ping)
15:19:29 Dong_Bu_Ring 802.1X[1809]:       Flags: 00
15:19:29 Dong_Bu_Ring 802.1X[1809]:       <reserved>: 0
15:19:29 Dong_Bu_Ring 802.1X[1809]:       Session ID: 16 (0x10)
15:19:29 Dong_Bu_Ring 802.1X[1809]: ->      Integer: 16 (0x10)
15:19:29 Dong_Bu_Ring BGP[1846]: AgentX: ping, Operational state, fail 0
15:19:29 Dong_Bu_Ring BGP[1846]: AgentX: pinging:
15:19:29 Dong_Bu_Ring BGP[1846]: AgentX: build Ping-PDU
15:19:29 Dong_Bu_Ring BGP[1846]: ->  AgentX Header:
15:19:29 Dong_Bu_Ring BGP[1846]:       Version: 1
15:19:29 Dong_Bu_Ring BGP[1846]:       Type: 13 (Ping)
15:19:29 Dong_Bu_Ring BGP[1846]:       Flags: 00
15:19:29 Dong_Bu_Ring BGP[1846]:       <reserved>: 0
15:19:29 Dong_Bu_Ring BGP[1846]:       Session ID: 15 (0x0F)
15:19:29 Dong_Bu_Ring BGP[1846]: ->      Integer: 15 (0x0F)
```

Why this occurs

In release 5.2.2: the **no snmp-server** command does not actually disable SNMP, it just de-configures SNMP so it is not available via the network. The SNMP software still continues to run and gather information from the protocol modules in the software. Even if SNMP appears to be disabled, the AgentX polling [as shown above] continues every 15 seconds.

The solution

In 5.3.1 and later releases: when SNMP is disabled, the connections between subagents and the master are broken. The lack of connections prevents the AgentX polling that would otherwise cause the CPU spikes.

In summary, this is expected behaviour in 5.2.2, and was fixed in 5.3.1.

MTR switch drops packets

Introduction

My Trace Route (MTR) combines the functionality of the traceroute and ping programs into a single network diagnostic tool. This tool investigates the network connection between the host on which MTR is running, and a user-specified destination host.

The issue

MTR does not report packet loss when directed to an AlliedWare Plus switch. However, it reports very high packet loss when directed to a device beyond the switch.

Why this occurs

To understand why this occurs, it is important to understand how MTR works. The MTR website is misleading. It states "it sends a sequence ICMP ECHO requests to each one", which is not strictly true.

What we have observed is that MTR sends two frames per100ms. These two frames are ICMP echo requests, destined for 192.168.1.2. One has a Time-To-Live (TTL) of 1 and the other has a TTL of 2.

So, MTR is sending ICMP echo requests, destined for the final hop, but with decreasing TTL values. This means that routers along the path will respond with ICMP Time-To-Live Exceeded messages, instead of ICMP echo replies.

This is significant because many network equipment vendors limit the rate of ICMP messages that are generated.

Further information about ICMP rate limiting in the Linux Kernel is available at:

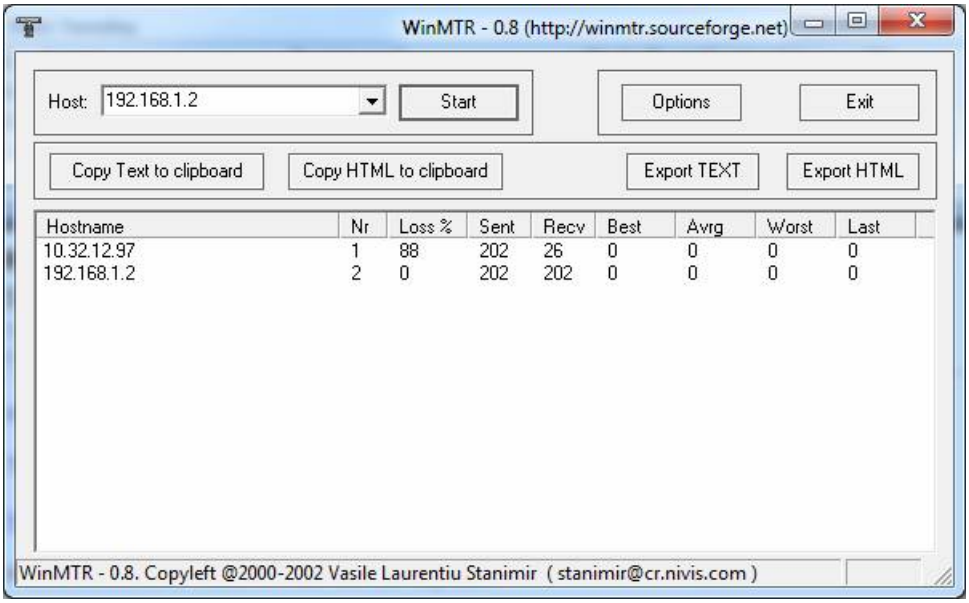
http://www.kernel.org/doc/man-pages/online/pages/man7/icmp.7.html

Example

1. The network is like this:

```
Pc1 (10.32.12.99) ---------- (port1.1.1, vlan1, 10.32.12.97) x908
(port1.1.2, vlan2, 192.168.1.1) ------------ (192.168.1.2, port49)
8648/2SP
```

2. When you run this test (ICMP packets at 100ms interval), MTR to 192.168.1.2 shows enormous packet loss:



3. MTR sends two frames every 100ms. These are ICMP echo requests, destined for 192.168.1.2. One has a TTL of 1 and the other has a TTL of 2.
4. The packet with a TTL of 2 reaches its destination and an ICMP echo reply is sent. This is correct.
5. The packet with a TTL of 1 reaches the AlliedWare Plus switch (10.32.12.97) and the TTL expires, so the switch sends back an ICMP Time-to-live Exceeded message. This is also correct.

   However, the Linux kernel employs ICMP rate limiting for certain ICMP packets. This means that only around 1 in 10 TTL expired packets will actually result in an ICMP Time-to-live Exceeded message in this scenario. This explains why MTR reports about 88% loss to the SwitchBlade x908.
6. In AlliedWare Plus, ICMP Echo Replies are not subject to the same rate limiting, which explains why when MTR is directed to the AlliedWare Plus switch, the rate of response is high. This is expected behaviour and is designed to prevent ICMP DoS attacks.

In summary

ICMP Time-To-Live Exceeded messages are rate-limited from an AlliedWare Plus switch, and ICMP Echoes are not rate limited. This explains the differences in behaviour.

The output example below shows the network traffic generated by MTR and the associated responses from the devices in the network. In this example, you can see that MTR sends two ICMP echo requests to 192.168.1.2. One has a TTL of 1, and one has a TTL of 2.

You can also see the ICMP time exceeded in-transit, with messages occurring approximately every 1 in 10 requests. This is the AlliedWare Plus Kernel rate-limiting the ICMP responses, and is the reason for the loss reported by MTR.

Note: This behaviour differs from the explanation on the MTR website. The website states that ICMP echo requests are sent to each host. This is not strictly true as shown by the capture below, where all ICMP Echoes are directed at the far end host, but with differing TTLs that will expire in transit and trigger a response from all L3 devices on route to the destination.

```
16:20:16.892634 IP (tos 0x0, ttl 1, id 45630, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 32514, length 44
16:20:16.943286 IP (tos 0x0, ttl 2, id 45631, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 32770, length 44
16:20:16.943975 IP (tos 0x0, ttl 63, id 17755, offset 0, flags [none], proto ICMP (1), length 64) 192.168.1.2 > 10.32.12.99:
  ICMP echo reply, id 60967, seq 32770, length 44
16:20:16.996321 IP (tos 0x0, ttl 1, id 45632, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 33026, length 44
16:20:16.997926 IP (tos 0xc0, ttl 64, id 36459, offset 0, flags [none], proto ICMP (1), length 92) 10.32.12.97 > 10.32.12.99:
  ICMP time exceeded in-transit, length 72
        IP (tos 0x0, ttl 1, id 45632, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2: ICMP echo
  request, id 60967, seq 33026, length 44
16:20:17.047216 IP (tos 0x0, ttl 2, id 45633, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 33282, length 44
16:20:17.047976 IP (tos 0x0, ttl 63, id 17756, offset 0, flags [none], proto ICMP (1), length 64) 192.168.1.2 > 10.32.12.99:
  ICMP echo reply, id 60967, seq 33282, length 44
16:20:17.100293 IP (tos 0x0, ttl 1, id 45634, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 33538, length 44
16:20:17.151302 IP (tos 0x0, ttl 2, id 45635, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 33794, length 44
16:20:17.151976 IP (tos 0x0, ttl 63, id 17757, offset 0, flags [none], proto ICMP (1), length 64) 192.168.1.2 > 10.32.12.99:
  ICMP echo reply, id 60967, seq 33794, length 44
16:20:17.204249 IP (tos 0x0, ttl 1, id 45636, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 34050, length 44
16:20:17.255502 IP (tos 0x0, ttl 2, id 45637, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 34306, length 44
16:20:17.257006 IP (tos 0x0, ttl 63, id 17758, offset 0, flags [none], proto ICMP (1), length 64) 192.168.1.2 > 10.32.12.99:
  ICMP echo reply, id 60967, seq 34306, length 44
16:20:17.308258 IP (tos 0x0, ttl 1, id 45638, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 34562, length 44
16:20:17.359319 IP (tos 0x0, ttl 2, id 45639, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 34818, length 44
16:20:17.360990 IP (tos 0x0, ttl 63, id 17759, offset 0, flags [none], proto ICMP (1), length 64) 192.168.1.2 > 10.32.12.99:
  ICMP echo reply, id 60967, seq 34818, length 44
16:20:17.412669 IP (tos 0x0, ttl 1, id 45640, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 35074, length 44
16:20:17.463376 IP (tos 0x0, ttl 2, id 45641, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 35330, length 44
16:20:17.463989 IP (tos 0x0, ttl 63, id 17760, offset 0, flags [none], proto ICMP (1), length 64) 192.168.1.2 > 10.32.12.99:
  ICMP echo reply, id 60967, seq 35330, length 44
16:20:17.516289 IP (tos 0x0, ttl 1, id 45642, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 35586, length 44
16:20:17.567295 IP (tos 0x0, ttl 2, id 45643, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 35842, length 44
16:20:17.567988 IP (tos 0x0, ttl 63, id 17761, offset 0, flags [none], proto ICMP (1), length 64) 192.168.1.2 > 10.32.12.99:
  ICMP echo reply, id 60967, seq 35842, length 44
16:20:17.620316 IP (tos 0x0, ttl 1, id 45644, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 36098, length 44
16:20:17.671299 IP (tos 0x0, ttl 2, id 45645, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 36354, length 44
16:20:17.671989 IP (tos 0x0, ttl 63, id 17762, offset 0, flags [none], proto ICMP (1), length 64) 192.168.1.2 > 10.32.12.99:
  ICMP echo reply, id 60967, seq 36354, length 44
16:20:17.724297 IP (tos 0x0, ttl 1, id 45646, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 36610, length 44
16:20:17.775392 IP (tos 0x0, ttl 2, id 45647, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 36866, length 44
16:20:17.775987 IP (tos 0x0, ttl 63, id 17763, offset 0, flags [none], proto ICMP (1), length 64) 192.168.1.2 > 10.32.12.99:
  ICMP echo reply, id 60967, seq 36866, length 44
16:20:17.828298 IP (tos 0x0, ttl 1, id 45648, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 37122, length 44
16:20:17.879354 IP (tos 0x0, ttl 2, id 45649, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 37378, length 44
16:20:17.879989 IP (tos 0x0, ttl 63, id 17764, offset 0, flags [none], proto ICMP (1), length 64) 192.168.1.2 > 10.32.12.99:
  ICMP echo reply, id 60967, seq 37378, length 44
16:20:17.932369 IP (tos 0x0, ttl 1, id 45650, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 37634, length 44
16:20:17.983342 IP (tos 0x0, ttl 2, id 45651, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 37890, length 44
16:20:17.984007 IP (tos 0x0, ttl 63, id 17765, offset 0, flags [none], proto ICMP (1), length 64) 192.168.1.2 > 10.32.12.99:
  ICMP echo reply, id 60967, seq 37890, length 44
16:20:18.036551 IP (tos 0x0, ttl 1, id 45652, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 38146, length 44
16:20:18.037920 IP (tos 0xc0, ttl 64, id 36460, offset 0, flags [none], proto ICMP (1), length 92) 10.32.12.97 > 10.32.12.99:
  ICMP time exceeded in-transit, length 72
        IP (tos 0x0, ttl 1, id 45652, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2: ICMP echo
  request, id 60967, seq 38146, length 44
16:20:18.087043 IP (tos 0x0, ttl 2, id 45653, offset 0, flags [none], proto ICMP (1), length 64) 10.32.12.99 > 192.168.1.2:
  ICMP echo request, id 60967, seq 38402, length 44
```

Hardware

Switch PSU fault analysis

Note: This issue applies to the SwitchBlade x908 switch only.

Introduction

The SwitchBlade x908 Switch provides AC and DC Power Supply options, and allows for Power Supply redundancy by providing two PSU Bays in the chassis. The PSUs are designed to provide long life and reliability. The units also provide good status and alarm communication for both monitoring and abnormal state information.

The AlliedWare Plus Operating System can provide PSU status information via either the CLI or SNMP MIBs. It has also been designed to extract as much information as possible when an alarm signal interrupt is received.

If a PSU becomes faulty, the micro-controller on the PSU may quickly decide to shut down, which means that information about what initiated failure can get lost. For this reason, AlliedWare Plus takes a snapshot of the information available as quickly as possible, once it receives an interrupt signal from the PSU. However, in the case of rapid shutdown, it cannot guarantee to capture the initial cause of the fault. Even so, the correct cause condition is usually stated, or can be deduced, as explained later in this article.

This Tips and Tricks item will aid in analysing and understanding any SwitchBlade x908 PWR05 PSU failures. The following sections will explain the types and meaning of information available from the PSU units, and explain about the variable results that can occur for given cause conditions.

Feature requirements

The ability to interrogate the I2C bus, to find an error code for logging after a PSU Indication Pin interrupt event, was introduced in Software Release 5.3.3-03. The error logging facility is important for PSU troubleshooting, therefore upgrading to this release or later is recommended.

PSU models this document applies to:

There are two main PWR05 variants, an AC and a DC version. The table below indicates the names used:

VERSION	ALLIED TELESIS MODEL NAME	MANUFACTURER'S MODEL REFERENCE
AC Version	AT-PWR05-AC	FNP600-12S153G
DC Version	AT-PWR05-DC	FND850-12DRG or FND850-12DRS101G

Information types and their meanings

Indication pins

There are a set of indication pins that the PSU uses to communicate:

- Device Present
- PSU Fan/Temperature Fault
- PSU Power Output
- PSU Power Input

Note: These indication pin values are also visible when viewing show system environment.

Interrogation of PSU I²C device

The switch CPU can seek data from the PSU via the I²C bus. The data is sought in response to the show system psu command, or in response to interrupts due to state changes on the PSU's Fan /Temperature Fault indication pin. In the case of an interrupt, the information is presented as an Error Code in the switch's system log.

How information is presented to the user

After a PSU interrupt event, the show log will log a single octet Error code. This code is in fact the first (most significant) octet of the Fault Bytes.

Here is an example of the show log output:

```
awplus#01:15:24 awplus HPI: SENSOR PSU slot 2 - PSU Power Output: BAD
01:15:24 awplus HPI: SENSOR PSU slot 2 - PSU Fan/Temperature Fault: BAD
- Error code 0x10
01:15:24 awplus HPI: SENSOR PSU slot 2 - PSU Power Output: BAD
```

The **show system psu** command quotes a two octet Fault Bytes figure in the Dynamic Data section, as shown below:

```
x908#sh sys psu
System PSU Information

Resource ID: 7  Name: AT-PWR05-AC   Bay: 2
  Part Number     : FNP600-12S153G
  Serial Number   : 080732-004PN
  Revision        : AA
  Mfg. date       : 2008-03-17
  Manufacturer    : POWER-ONE
  Mfg. location   : 2

  Device Ratings:
  Output rail 1   : 12000 mV, 51000 mA
  Output rail 2   : 12000 mV, 500 mA
  Output Power    : 606 W
  Min AC input    : 90 V
  Max AC input    : 264 V

  Dynamic Data:
  Fault Bytes     : 21 01
  Time in service : 3946 hours
  Measured rail 1 : 0 mV, 0 mA
x908#
```

SNMP Traps

PSU Temperature and Fan Alarms also produce SNMP Trap events based on the AT-ENVMONv2-MIB. Information about this MIB is available in the SNMP MIBs chapter of the SwitchBlade x908 and x900 Series Switches AlliedWare Plus Operating System Software Reference:

http://www.alliedtelesis.com/support/documentation

About these examples:

- The Error Code shows 0x10, meaning Temperature-Prewarning. However, we know that the PSU only actually alarms (causes interrupt) when the Over-Temperature threshold is reached, therefore the code should have indicated 0x20. In this case, on interrupt the CPU has actually probed the PSU I2C device before the Fault Byte bits were changed. For example, the PSU can ometimes send an interrupt a while before it alters its I2C bus fault bytes. If this happens, the before interrupt error code 0x10 is displayed.
- If a redundant PSU is still operational, after this PSU thermal failure the Fault Bytes show a realistic end-result figure of 0x 21 01 - this means Over Temperature, Power Supply NOT OK, and Output 1 Voltage Not OK.

Meaning of the show system PSU Fault Bytes

The PSU's I²C device expresses alarm states by setting individual bits within the Fault Bytes.

The following example shows the make-up of the two octets, and defines the bit positions of the significant alarms:

```
Two Bytes Position Numbers:
<< MSB        LSB >>
76543210  76543210
```

Fault Byte 1
(The Error Code Octet)

Bit Position /Meaning:
7 -
6 - Fan Not OK
5 - Over Temperature
4 - Temperature Pre-warning
3 -
2 -
1 - AC Not In range
0 - Power Supply NOT OK

Fault Byte 2
(This Octet is NOT quoted as part of Error code. It is miscellaneous information).

Bit Position /Meaning:
7 -
6 -
5 -
4 - Output 1 Current NOT OK
3 -
2 -
1 -
0 - Output 1 Voltage Not OK

Because alarm states are expressed by setting individual bits within the error byte, several alarms can be enabled simultaneously. A fault condition only has a distinct hex value if it is the only alarm active. If there are other faults, then the hex value is the sum of both fault values.

The original cause value is often only available for inspection for no longer than 1 second. This is why the alarm code quoted in the log is not always the cause code, and why the command **show system psu** often only shows a PSU shut-down status, rather than the cause condition.

Meaning of the show log error codes

As previously mentioned, when the PSU Fan /Temperature Fault indication pin changes state, this causes an interrupt to the switch's CPU, which then in turn interrogates the PSU's I^2C for further information.

This information is displayed in a log message, and quotes a single octet error code. This single octet is in fact the first, or most significant octet of the Fault Bytes discussed above.

When translated to Hex values, the initial distinct error code values of fault conditions are:

0x10 - Temperature Pre-Warning
0x20 - Over Temperature
0x40 - Fan Fail

Note that these are not necessarily the values that will be logged.

For example: For Over Temperature, the binary value of the first octet of the Fault Bytes will be - 00100000 - and this translates to a hex value of 0x20. However, for Power Supply NOT OK the Fault Bytes may be 00100001, which is a hex value of 0x21.

Understanding the variable data results

Both versions of the PWR05 PSU were tested to show the typical Error Code and Fault Bytes values that are logged in failure conditions. Because the values dynamically change at the moment of failure, the captured value is not always the expected initial value.

This can be because interrogation has happened too quickly, before bits have been set; or too late, after bits have been reset.

Here are the tested typical values:

MODEL	CAUSE FAULT CONDITION	ERROR CODE SHOULD BE	TYPICAL FINAL VALUE QUOTED IN LOG ERROR CODE	TESTED FINAL SHOW SYSTEM PSU FAULT BYTES VALUE
PWR05 AC	Fan Fail	0x40	0x40	0x 01 01
PWR05 AC	Thermal Failure	0x20	0x10	0x 21 01
PWR05 DC	Fan Fail	0x40	0x00	0x 41 00
PWR05 DC	Thermal Failure	0x20	0x20	0x 21 01

Notes:

- The tested final show system psu Fault Bytes value is the expected value assuming there is a redundant PSU still operational. To enter the command show system psu on the switch after a power supply shut-down, the switch must have a redundant PSU still operational. If the PSU that shut down was the only operational PSU in the switch, then shut down of the PSU would have shut down the whole switch.
- Thermal failure should indicate 0x20. For the AC model, you typically see the Temp Pre-warning value 0x10 instead.
- The Fan Fail should indicate 0x40. For the DC version, you typically see 0x00 instead, because the bit is not set in time.

How to determine a PSU failure cause when the log is inconclusive

As previously mentioned, the AlliedWare Plus Operating System does not always capture the initial cause of the fault. If no cause issue is shown, then you need to figure out if the failure was due to Fan Failure, or to Over Temperature.

If the failure was due to Over Temperature, then the temperature would have been climbing prior to the shutdown event. As the temperature climbed, other sensors in the switch would have indicated some temperature events.

Therefore:

- If there were prior temperature alarms elsewhere in the switch, then it was caused by over-temperature. This is often caused by high ambient /room temperature.
- If not, it was caused by fan failure.

The SwitchBlade x908 directs air through the chassis first, and then through the PSU. Therefore in the case of high ambient temperatures, any over-temperature failure would be pre-warned by switch chassis or module temperature alarms.

While the PSU has a temperature pre-warning fault code, this state does not initiate an alarm state on the indication pins, therefore the PSU pre-warnings are not logged.

Temperature operating range

Allied Telesis Lab testing has shown that the PWR05 AC version can tolerate ambient air temperatures of up to 72 - 84 degrees C before tripping, for an AC supply of either 110v or 230v.

Official manufacturer documentation indicates a more conservative trip point as follows:

Over-temp set point: 71.5degrees C
Recovery temp: 65.5degrees C

Fault sequences

The PSU micro-controller fault sequence

1. PSU detects a fan fail or over-temperature condition.
2. PSU changes the indication pin for PSU Fan/Temperature Fault (causing an interrupt to the SBx908), and lights the PSU O/T LED.
3. PSU shuts the PSU output down, changes the indication pin for output power and extinguishes the PSU Power Out LED.

The SwitchBlade x908 CPU fault sequence

1. CPU receives an interrupt indicating that a PSU indication pin has changed state.
2. CPU retrieves the PSU indication pin states from the SBx908 PSU monitor.
3. CPU interrogates the PSU's I^2C device to get the Fault Bytes.
4. CPU takes appropriate action to indicate the fault.

During a fault condition, the PSU Micro-Controller first commences its 3-step fault sequence. When PSU event #2 occurs, the SwitchBlade x908 begins its fault sequence.

The timing of PSU event #3 may fall at any point during the switch's fault sequence. If you are lucky, then PSU event #3 does not occur until the end of the SwitchBlade x908 fault sequence. But if PSU event #3 occurs somewhere in the middle of the SwitchBlade x908

fault sequence, the amount of information that the SwitchBlade x908 can present to the user about the cause of the error is unpredictable.

PSU Checksum and Serial Number Corruption

It is very important to install and handle PSU units correctly. If not, you may see see corruption of the PSU EEPROM information. In every case this is because the PSU unit is either plugged into the x908 chassis while it is powered on, or because the PSU unit was powered up outside of the chassis - meaning that it was not properly earthed.

Example: If the PSU EEPROM has become corrupted, it can lead to information like this:

```
show system...
PSU         298   PSU2   AT-PWR05-AC
A-0   PSU read fail
show system psu
================================================================
Resource ID: 7   Name: AT-PWR05-AC    Bay: 2
 The checksum of the information read from this PSU is incorrect.
 The information below is the data that was read, but may have errors.
 Part Number      :
 Serial Number    :
 Revision         :
 Mfg. date        : 2000-00-00
 Manufacturer     :
 Mfg. location    : 00
 Device Ratings:
 Output Power     : 0 W
 Min AC input     : 0 V
 Max AC input     : 0 V
 Dynamic Data:
 Dynamic data invalid. PSU may be powered off.
```

Best practice PSU handling

To avoid EEPROM data corruption, always use best practice for inserting PSUs:

1. Schedule a short outage of the switch
2. Power down the x908 chassis
3. Insert the new PSU Unit and ensure that the unit has been properly plugged-in
4. Power up, and check the show system psu information.

This practice ensures that the I2C bus that is used in the SwitchBlade x908 to read the PSU EEPROM can be read correctly at start-up, because it ensures the PSU is properly earthed. Good earthing also avoids permanent EEPROM data corruption.

What to do if your PSU EEROM data is corrupted

If your EEROM data is corrupted, it may be a temporary or permanent data corruption. Temporary corruption occurs only because the I2C bus was not able to read correctly at power up time because of the way the unit was plugged in. Therefore, try again. Schedule a short outage of the switch, power down the chassis, ensure that the PSU units are properly plugged into the chassis, wait several seconds then power up again.

Permanent EEPROM data corruption can also occur due to bad earthing. You are more vulnerable to this happening if the PSU has been powered up while outside of the chassis. If this has occurred, the data cannot easily be corrected, but in most cases it does not affect the

PSU's performance of the PSU other than to have corrupted a section of your displayed data when you use the show command.

Note: A PSU design improvement was made from revision Rev AH to help minimise the risk of EEROM reading or data corruption.

Addendum

Information about upcoming POE supply.

At the date of publication, the Power over Ethernet (PoE) version of the PSU had not been released. Early indications are that this PSU will not have Fault Byte information available. It will only have a simple alarm supplied via Indication Pins.

C613-16113-00 REV D

Allied Telesis the solution : the network

North America Headquarters | 19800 North Creek Parkway | Suite 100 | Bothell | WA 98011 | USA | T: +1 800 424 4284 | F: +1 425 481 3895
Asia-Pacific Headquarters | 11 Tai Seng Link | Singapore | 534182 | T: +65 6383 3832 | F: +65 6383 3830
EMEA & CSA Operations | Incheonweg 7 | 1437 EK Rozenburg | The Netherlands | T: +31 20 7950020 | F: +31 20 7950021

alliedtelesis.com