

Technical Tips and Tricks | AlliedWare Plus™ Managed Layer 3 Switches

Introduction

This document contains useful technical tips and tricks for AlliedWare Plus Managed Layer 3 switches.

These Tips and Tricks apply to:

SwitchBlade™ x908
x600-24Ts
x600-24Ts-POE
x600-24Ts/XP
x600-48Ts
x600-48Ts/XP
x900-12XT/S
x900-24XT
x900-24XT-N
x900-24XS

Contents

1.	Management	2
	Using shell scripts	2
	Using command scripts	4
	Large configurations in the "conf t" environment	6
	Using SFTP to transfer files to/from an AlliedWare Plus Switch	7
2.	Switching	13
	How to view switch tables	13
	How to view port counters	14
	Adding a VLAN to an LACP trunk causes port flapping	15
	IGMP Proxy appears to only partially work	16
	MTU/MRU command changes	21
	Using QoS to mark packets' DSCP value, and assign them to a queue	24
	Unable to use the multicast address 232.x.x.x without specifying a source	26
3.	Resiliency	27
	x600 resiliency link	27
	The reboot rolling command	31
	The remote-login command	33
	The show license command	34
	Provisioning	36
4.	Diagnostics	42
	CPU usage spikes	42
	MTR switch drops packets	45
5.	Hardware	48
	Switch PSU fault analysis	48

Management

Using shell scripts

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

Important: 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 figure 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. These are similar to scripts in AlliedWare.

Command scripts must not have the file extension `.sh`. We recommend using `.scp`.

Note: *Command scripts are different to device configuration files.*

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
```

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

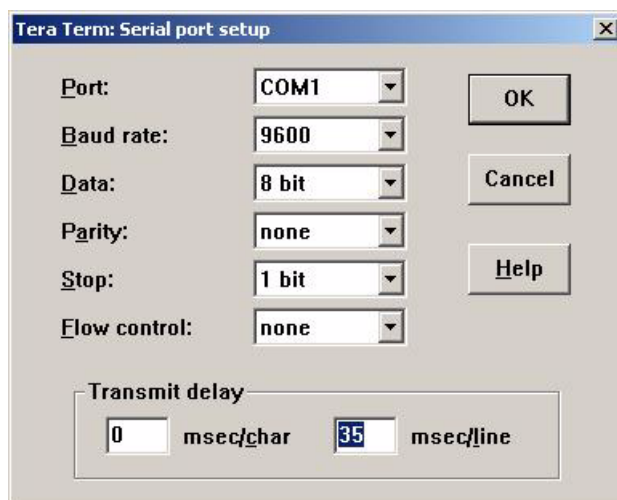
Pasting in very large configurations using the console at the "conf t" prompt can give unpredictable results. Consoles, as standard practice, do not have flow control. If too much text is pasted, it will exhaust the buffer size available for the console.

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.
```

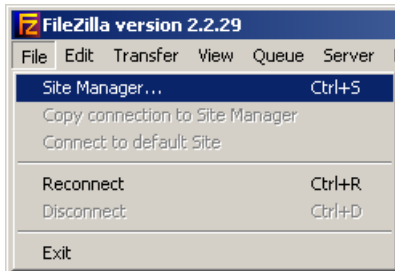
3. Enable one or more users to access SSH:

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

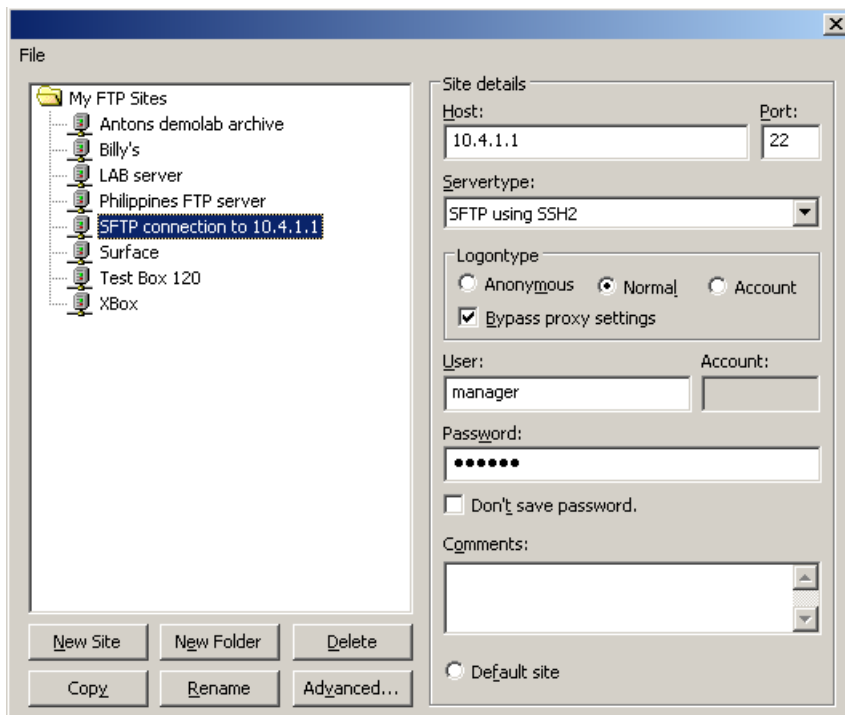
Configuring Filezilla

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

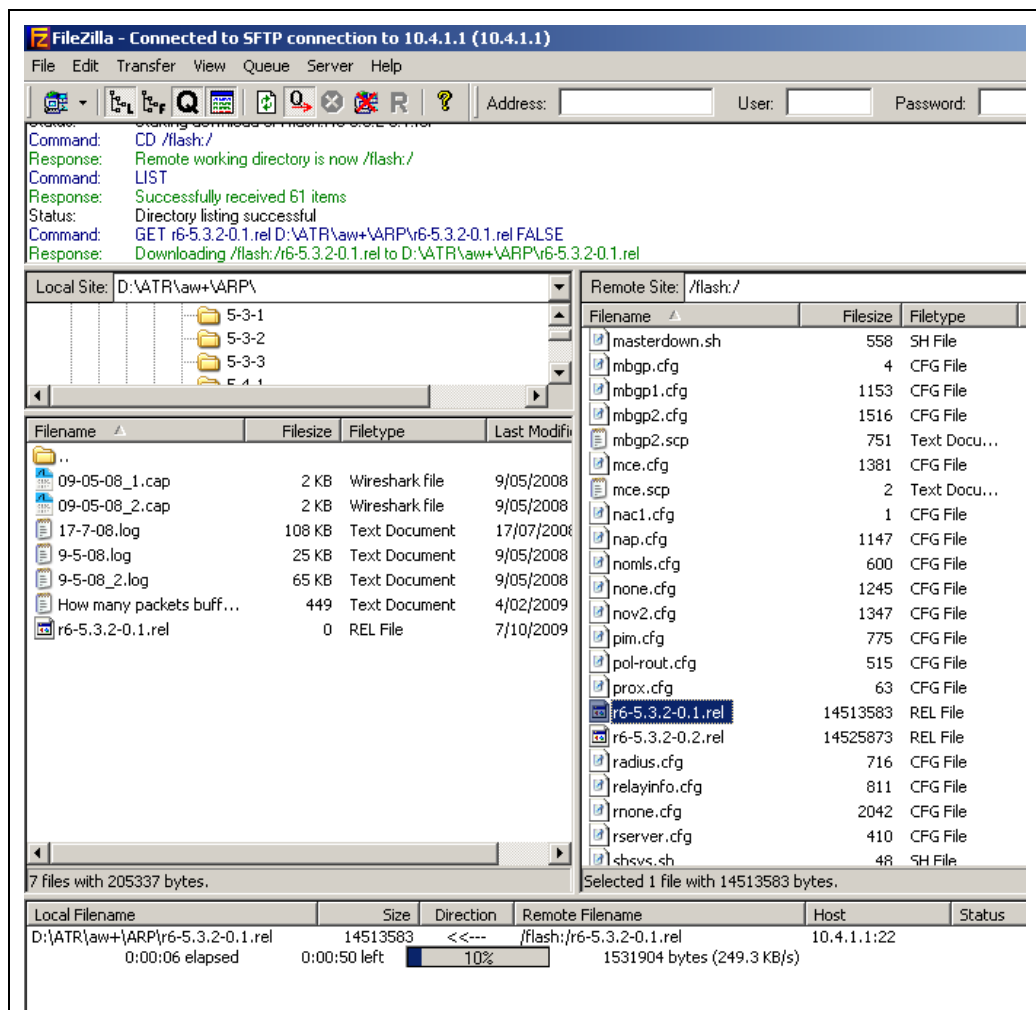
1. Select **File > Site Manager...**



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.

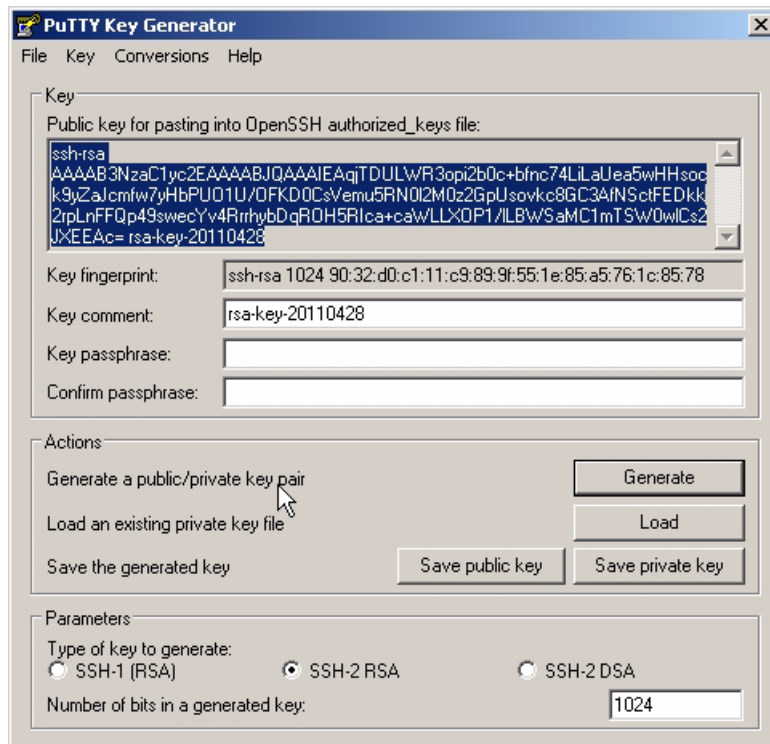
This example uses the Putty suite of secure device management and file transfer tools. You can download these tools, puttygen.exe, psftp.exe and pscp.exe, at:

<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 puttygen.

1. In puttyGen, 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 2 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:

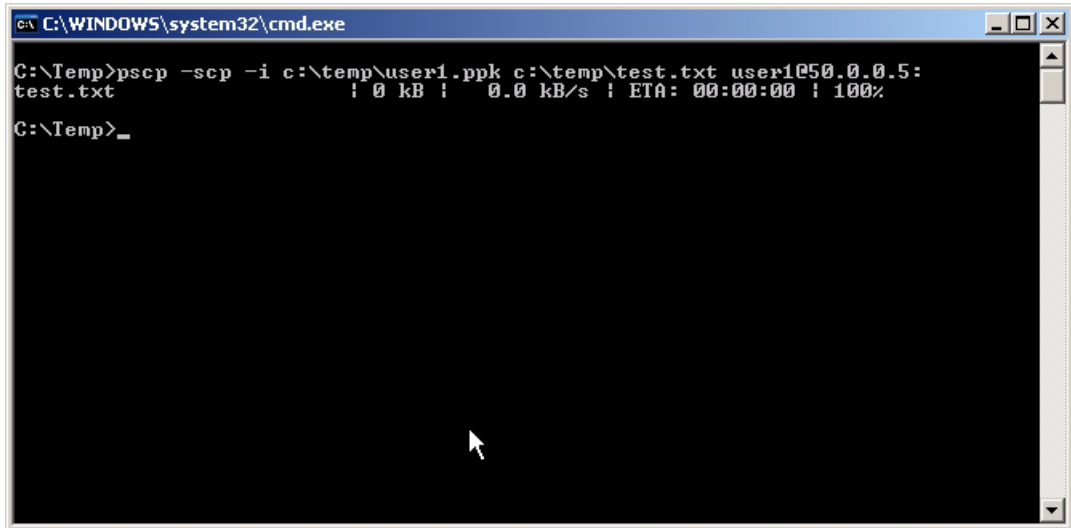

```
awplus(config)#crypto key pubkey-chain userkey user1 user1.pub
```

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:



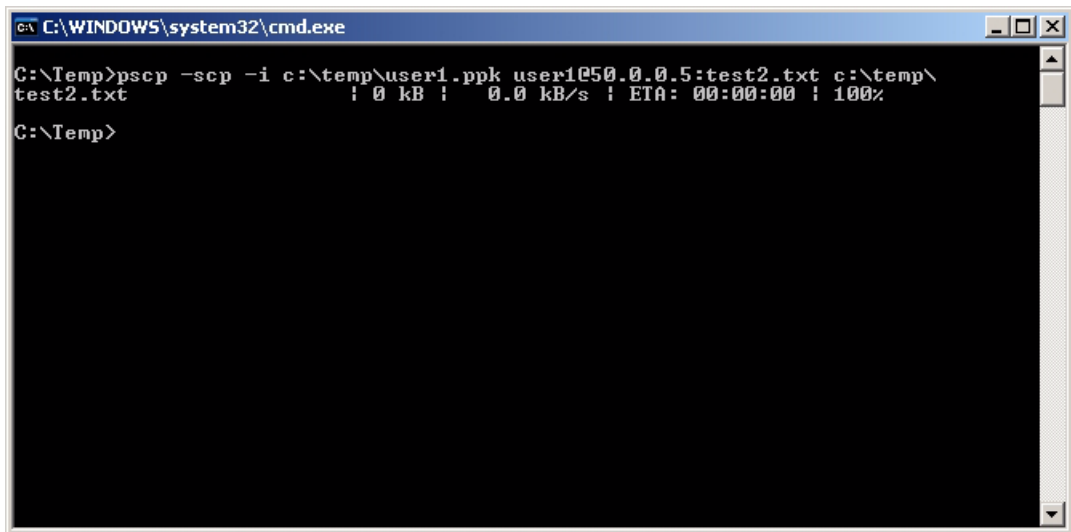
The command:

```
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

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

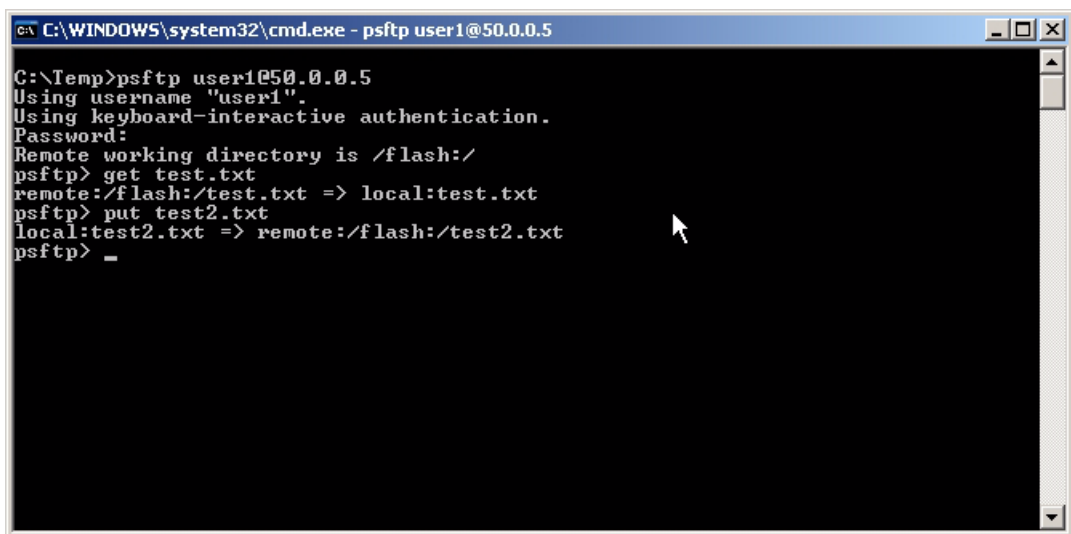


```
C:\WINDOWS\system32\cmd.exe
C:\Temp>pscp -scp -i c:\temp\user1.ppk user1@50.0.0.5:test2.txt c:\temp\
test2.txt
| 0 kB | 0.0 kB/s | ETA: 00:00:00 | 100%
C:\Temp>
```

```
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



```
C:\WINDOWS\system32\cmd.exe - psftp user1@50.0.0.5
C:\Temp>psftp user1@50.0.0.5
Using username "user1".
Using keyboard-interactive authentication.
Password:
Remote working directory is /flash:/
psftp> get test.txt
remote:/flash:/test.txt => local:test.txt
psftp> put test2.txt
local:test2.txt => remote:/flash:/test2.txt
psftp> _
```

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
```

The AlliedWare Plus port counters are similar to the AlliedWare 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.

IGMP Proxy appears to only partially work

Introduction

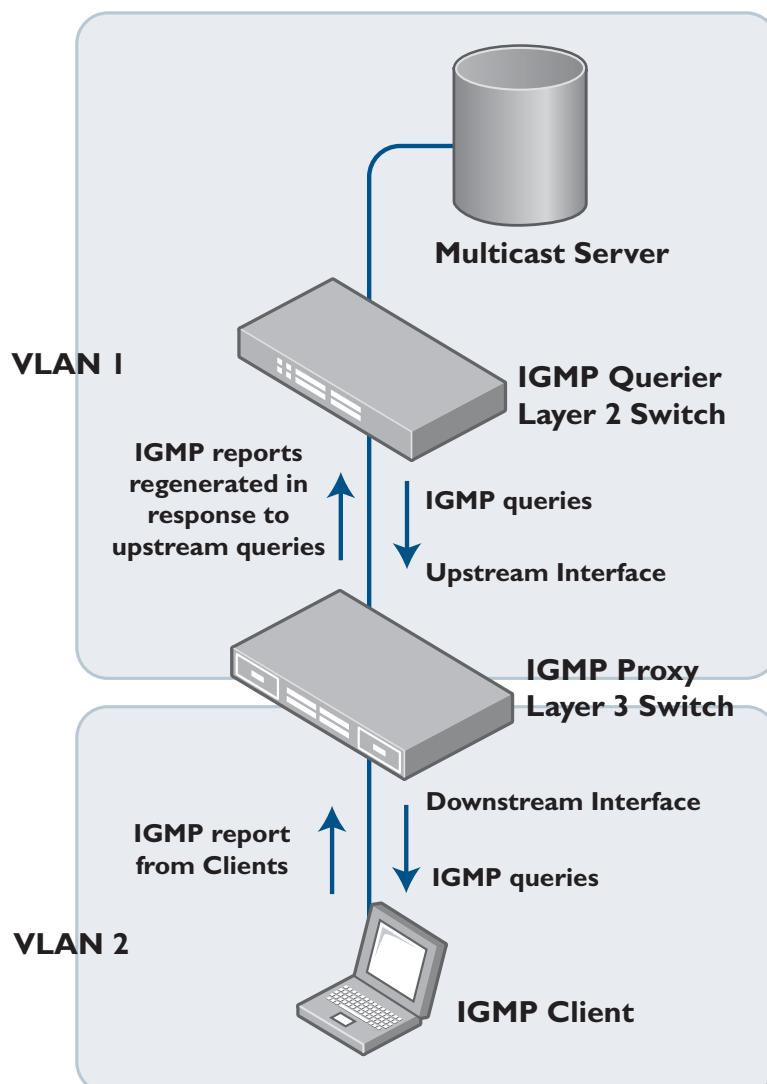
At times, IGMP Proxy users experience an issue where multicast streams to subscribers do not last more than four minutes. This occurs when IGMP Proxy roles are incorrectly deployed in the network topology.

IGMP Proxy: correct roles

In examining this issue, it is helpful to start with a look at correct deployment of IGMP Proxy. When configuring IGMP Proxy, it is important to bear in mind the entire IGMP network topology, and to have a clear understanding of which role has been given to each interface and/or device. Available roles are:

- Multicast Server - source of multicast streams
- IGMP Querier - one for each L2 broadcast domain
- IGMP Proxy Upstream Interface - emulates client behaviour
- IGMP Proxy Downstream Interface - may also be downstream IGMP Querier
- IGMP Client

This diagram shows the correct IGMP proxy roles:



The issue

In some situations, IGMP Proxy can appear to be working incorrectly, as multicast streams to subscribers do not last more than four minutes. This is because only the first join/report message is passed through the Proxy, and it **appears as though** IGMP Proxy fails to respond to the IGMP Queries it receives from upstream with subsequent reports. In fact, IGMP did not receive any queries, so had nothing to respond to.

To resolve this issue, you must ensure that the correct IGMP roles are given to the correct devices. In cases where the issue is experienced, usually the upstream interface of the IGMP proxy has also been configured as IGMP Querier on the same interface. This combination causes a conflict between the two roles.

When the upstream interface is configured with the conflicting roles: although the first (unsolicited) join/report message from a downstream client is passed through the Proxy, subsequent reports from clients are not immediately regenerated. This is correct behaviour. Instead, the IGMP Proxy on the upstream interface acts like a client itself, **on behalf of the downstream clients**. As such, it only responds with reports **after** receiving queries from upstream.

In this role conflict case, the IGMP Proxy upstream interface **does not receive** IGMP Queries arriving on the upstream interface. The queries are in fact being **sent by itself**. They are being **sent** on the upstream interface, but not **received** on the upstream interface. The IGMP Proxy cannot "hear" messages **from itself** as IGMP Querier. It does not believe it is being queried, therefore it does not see the need to respond with reports.

This is the role conflict that must be avoided to solve this issue.

Technical background of the issue

This issue is easier to understand when you re-examine all the activities that a correctly configured IGMP Proxy undertakes. The IGMP Proxy acts as though it were a Client on the upstream interface on behalf of the downstream Clients. As such, it:

1. Regenerates the initial (unsolicited) join/report message started from any **downstream** Client.
2. Forwards the multicast streams to the **downstream** interface. In conjunction with IGMP Snooping, the multicast stream is forwarded to the appropriate port where the Client resides.
3. Sends queries into the **downstream** interface L2 domain, if it is configured as IGMP Querier and if it wins Querier election.
4. Listens for and notes any report messages received from **downstream** Clients in response to query messages sent in that L2 broadcast domain. These reports confirm which multicast streams are still required by at least one downstream Client. Note that these reports are not simply forwarded upstream - the upstream interface needs to receive a query message first.
5. Listens for any queries about streams from **upstream** interfaces, and respond with reports if it is known that at least one downstream Client is still sending reports requesting that stream.

However, as previously discussed, the issue arises if the **upstream** interface of the IGMP Proxy device has been wrongly configured as Querier **as well as** Upstream Proxy interface, and if that interface becomes the elected Querier, then step 5 above causes a conflict. The IGMP Proxy does not hear itself as querier, and therefore does not send reports.

In this conflict configuration, you can observe the multicast stream time-out on the IGMP Querier using the command **show ip igmp groups**.

The Solution

Correct configuration and correctly nominating roles is important for a successful IGMP Proxy solution.

As shown in the diagram in “[IGMP Proxy: correct roles](#)” on page - 16, the correct roles are:

- A Multicast Server
- A separate IGMP Querier in the first broadcast domain (VLAN)
- The IGMP Proxy (L3 switch) is a member of two broadcast domains, and it has either two or three roles:
 - The IGMP Proxy Upstream Interface. This interface must **not** be querier.
 - The IGMP Proxy Downstream Interface
 - Optionally, the Downstream Interface may also be IGMP Querier.
- On the downstream VLAN, one or multiple IGMP Clients.

On the IGMP Proxy device, it is important to avoid an interface being configured to facilitate the Upstream IGMP Proxy role and IGMP Querier role simultaneously, in order to avoid the role conflict.

Correct configuration of an AlliedWare Plus IGMP Proxy solution

Note: This configuration example shows a simple Layer-2 switch with an IGMP Querying role.

Configuration of the Upstream IGMP Querier

```
! turn on multicast routing on the router
ip multicast-routing
! Optional line - IGMP Snooping is actually on by default
ip igmp snooping
interface vlan2
 ip address 1.1.3.2/24
! This enables IGMP Querier role
 ip igmp
```

Configuration of IGMP Proxy

```
! turn on multicast routing on the router
ip multicast-routing
! Optional line - IGMP Snooping is actually on by default
ip igmp snooping
! Upstream interface
interface vlan1
ip address 1.1.3.1/24
! This sets the IGMP Proxy upstream interface
ip igmp proxy-service
! IGMP Proxy *upstream* interface and the IGMP Querier role *cannot* co-exist.
! As shown above, for correct configuration it must ONLY define the proxy
  service.
!
! First downstream interface
interface vlan2
ip address 1.1.1.2/24
! This enables IGMP Querier role
ip igmp
! This sets the IGMP Proxy downstream interface.
ip igmp mroute-proxy vlan1
! IGMP Proxy *downstream* interface and the IGMP Querier role *can* and often
  will be configured together.
```

Inspecting operation of an IGMP proxy solution

You can confirm the behaviour of IGMP Proxy using the **show ip igmp groups** command. It is useful to check the subscribed streams on both the IGMP Proxy device and the upstream IGMP Querier device.

In a correctly working proxy, you can expect to see the time to expiry value for the stream being refreshed regularly to a larger value - around 4 minutes, and this confirms that reports have been received back for the queries issued. You can expect to see this happening on both the Upstream IGMP Querier device and the IGMP Proxy device:

Upstream IGMP Querier Device

```
IGMP Querier#show ip igmp groups
IGMP Connected Group Membership
Group Address      Interface          Uptime    Expires    Last Reporter
...
225.10.10.10      vlan2             00:03:50 00:00:30 1.1.3.1
```

Below you can see that time to expiry has been refreshed, therefore the report message was received to keep the stream alive:

```
IGMP Querier#show ip igmp groups
IGMP Connected Group Membership
Group Address      Interface          Uptime    Expires    Last Reporter
...
225.10.10.10      vlan2             00:03:57 00:03:27 1.1.3.1  < increased
  time-to-expiry
```

IGMP Proxy Device

```
IGMP Proxy#sh ip igmp groups
IGMP Connected Group Membership
Group Address      Interface      Uptime    Expires    Last Reporter
224.0.1.22         vlan1         00:00:32  stopped    0.0.0.0
225.10.10.10       vlan1         00:00:30  stopped    1.1.4.11
239.255.255.250    vlan1         00:00:36  stopped    0.0.0.0
224.0.1.22         vlan2         00:00:32  00:03:47  1.1.1.88
225.10.10.10       vlan2         00:00:25  00:03:54  1.1.1.88
```

On the downstream **vlan2**, note that the expiry time has increased between these two commands, meaning that a new report message was received.

Note: the proxy upstream interface lists the streams as stopped. This is normal behaviour because the upstream interface does not receive reports, it actually sends them.

```
IGMP Proxy#sh ip igmp groups
IGMP Connected Group Membership
Group Address      Interface      Uptime    Expires    Last Reporter
224.0.1.22         vlan1         00:00:41  stopped    0.0.0.0
225.10.10.10       vlan1         00:00:39  stopped    1.1.4.11
239.255.255.250    vlan1         00:00:45  stopped    0.0.0.0
224.0.1.22         vlan2         00:00:41  00:04:16  1.1.1.88
225.10.10.10       vlan2         00:00:34  00:04:16  1.1.1.88
                                     < increased time-to-expiry
```

MTU/MRU command changes

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 both x600 and x900 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 both x600 and x900 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 both x600 and x900 switches, you can only set the MTU on L3 interfaces (VLANs).
4. On both x600 and x900 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 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 the MTU for VLAN's whose members are non-trunked (do not belong to any other vlans).
6. On x600 switches, there is a new MRU command:

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

Note: The default of 16383 is consistent with previous releases. Also note that the maximum MRU is 16375 which is 16383 - 4 bytes for VLAN tag and - 4 bytes for CRC.

7. On x900 switches, enabling the Jumboframe setting sets the MRU of all ports to 10240 bytes (this is to make it consistent with previous releases). However enabling the Jumboframe setting does not automatically set the MTU
8. On x900 switches, you cannot set the MRU for individual ports.
9. On both x600 and x900 switches, the MRU setting is only shown for ports, and MTU settings are only shown for VLANs.
10. On x600 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 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 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

```

Tests

Command line tests

- boundary testing
- show command testing
- configuration command testing

Operational testings

- MRU tests on x600 switches (including jumboframe)
- MTU tests on x600 switches (including ICMP Too big observation)
- Jumboframe tests on x900 switches
- Non-jumboframe tests on x900 switches
- MTU test on x900 switches (including ICMP Too big observation)

Technical notes

- On x600 switches, setting the MRU for a port affects the MAXFR (bits 15:0) and EGR_MTU_SIZE (bits 13:0) registers for the port. For example, for port I.0.1 the register is MAXFR.ge0 and EGR_MTU_SIZE.ge0 (on unit 1)

```
sh plat reg all | grep MAXFR.ge0
```

```
sh plat reg all | grep EGR_MTU_SIZE.ge0
```

- On x600 switches, setting L3 MTU for a port affects the EGR_MTU_SIZE (bits 27:14) register for the port. For example, for port I.0.2 the register is EGR_MTU_SIZE.ge1 (on unit 1)

```
sh plat reg all | grep EGR_MTU_SIZE.ge1
```

- Recommended defaults by Broadcom:
 - MAXFR (15:0), 0x5ee (1518, allows tagged frame 1522)
 - EGR_MTU_SIZE:
 - L3_MTU_SIZE : (27:14), 0x3fff (16383)
 - MTU_SIZE : (13:0), 0x3fff (16383)
- On x900 switches, setting Jumboframe mode affects the Ports Control Register:
 - Unit x - ports 0,1,2,6,7,8 -> 0x00804004 (bits 8:6 for GE)
 - Unit x - ports 3,4,5,9,10,11 -> 0x01004004 (bits 8:6 for GE)

The values mean:

0 = max 1518	4 = max 9192
1 = max 1522	5 = max 10240
2 = max 1552	6 = max 2048
3 = max 9022	7 = reserved

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

This applies to x600 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 switch, DSCP values cannot be premarked in packets prior to policing. They can only be remarked after policing.

So, the command **police single-rate <cir> <cbs> <pbs> 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

So, 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 an 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.

Resiliency

x600 resiliency link

Note: *This issue applies to the x600 series switches only.*

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 2 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, the x600 switches don't have an out-of-band Ethernet port. So a resiliency link within an x600 VCStack 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 x600 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 x600 resiliency link

1. Once the x600 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#sh 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 no more than one unit of the stack is rebooting at any given time.

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

```
awplus#sh 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
4pDI724ugtNcqlf8BmZMtI2YEX6MS1S0GxDGCSlaf8aAYVDz
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 preconfigure 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 preconfigure 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 ports 1.1.1-1.1.12, 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
```

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. Note that 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 ports 1.0.1-1.0.12 is the default port configuration. The port trunk configuration provisioned for the XEM-IXP 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 could 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-IXP 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 had instead removed the provisioned XEM and added another, the overlapping port (port2.2.1) would have been deleted and any configuration on it lost:

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

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 **tracert** 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 per 100ms. 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:

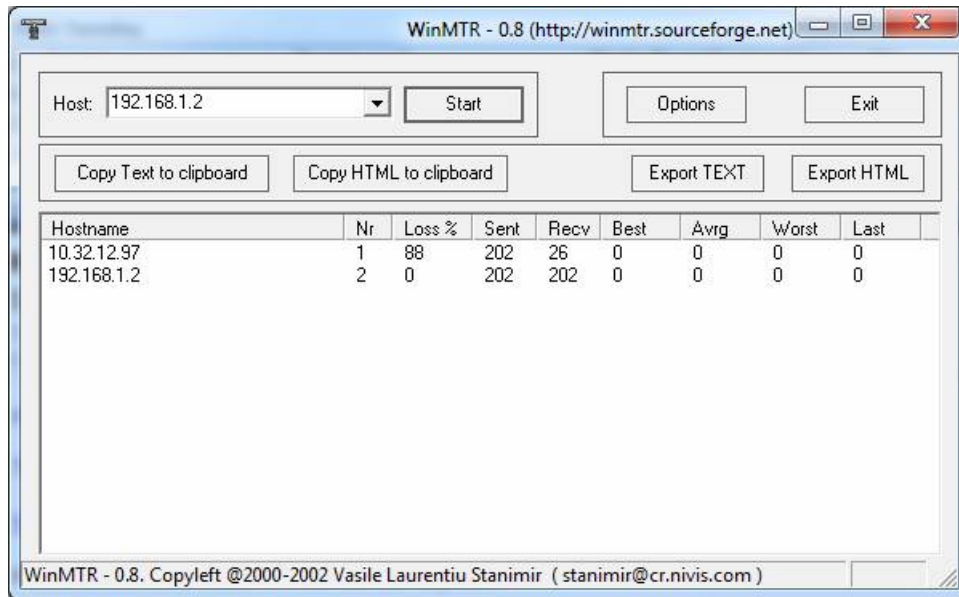
<http://www.kernel.org/doc/man-pages/online/pages/man7/icmp.7.html>

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 Names	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/media/datasheets/reference/x900_alliedware_plus_ref_a_v532.pdf

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 sometimes 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 I 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 I Current NOT OK
- 3 -
- 2 -
- 1 -
- 0 - Output I 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 **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 I²C 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 I²C 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 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.

USA Headquarters | 19800 North Creek Parkway | Suite 200 | Bothell | WA 98011 | USA | T: +1 800 424 4284 | F: +1 425 481 3895
European Headquarters | Via Motta 24 | 6830 Chiasso | Switzerland | T: +41 91 69769.00 | F: +41 91 69769.11
Asia-Pacific Headquarters | 11 Tai Seng Link | Singapore | 534182 | T: +65 6383 3832 | F: +65 6383 3830
www.alliedtelesis.com

© 2011 Allied Telesis Inc. All rights reserved. Information in this document is subject to change without notice. All company names, logos, and product designs that are trademarks or registered trademarks are the property of their respective owners.

C613-16113-00 REV B

Connecting The  World

 Allied Telesis