How To | Troubleshoot Slow Network Performance

Introduction

This How To Note describes techniques for troubleshooting slow network performance. When troubleshooting a network slow down problem, it is very important to keep in mind that hardware-based forwarding in switches never slows down - it continues to operate at wire speed on all ports all the time. Therefore, the cause of slow network performance is never going to be something that is causing a slow down of hardware switching.

In fact, the three most common reasons for a network slow down are:

- congestion
- data corruption
- collisions

A fourth, less common, cause for network slow down is:

- a high rate of STP Topology Change Notifications.

This document provides useful strategies to help you methodically investigate and resolve the issues that may be slowing your network down.

Which products does it apply to?

This document applies to the following Allied Telesis routers and managed Layer 3 switches:

- SwitchBlade x8100
- SwitchBlade x908
- x600 and x610 series switches
- x900 series switches
- x510 series switches

Network Slowdown

Typically, slow performance is not experienced across a whole network. Rather, most often what happens is that some or all workstations on the network experience slow performance of applications interacting with one, or a set, of network servers.

Before looking into the network switches for the cause of the slowdown, it is important to first verify that the slow performance is not due to resource overload on the servers or workstations themselves.

If there is good evidence that the cause of the slow performance lies in the switching infrastructure, rather than the workstations or servers, then the next step in troubleshooting is to identify one or more network paths that are experiencing frequent occasions of slow performance. Identify particular workstations and particular servers between which communication is frequently slow, and map out the network path between them - i.e. which ports of which switches do the packets exchanged between the workstation and server pass through?

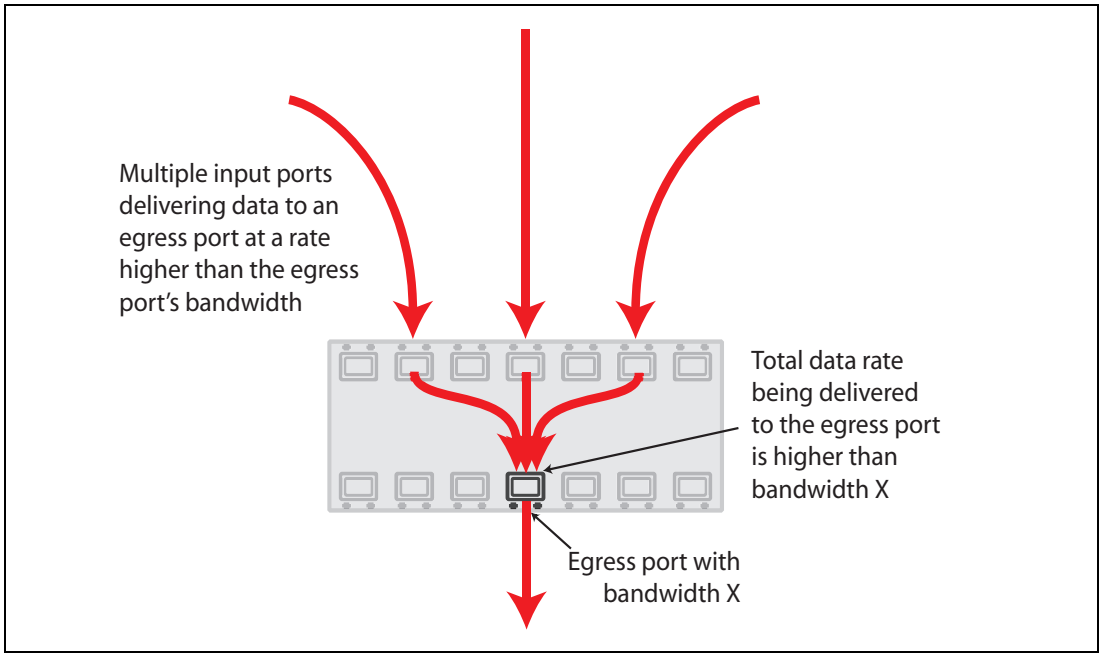
Then, work along the identified network path(s) looking for evidence of congestion, corruption, or collisions, as described in this How To Note.

Congestion

The first task in debugging a network slow down is to look for ports that are **oversubscribed**.

Oversubscribed ports

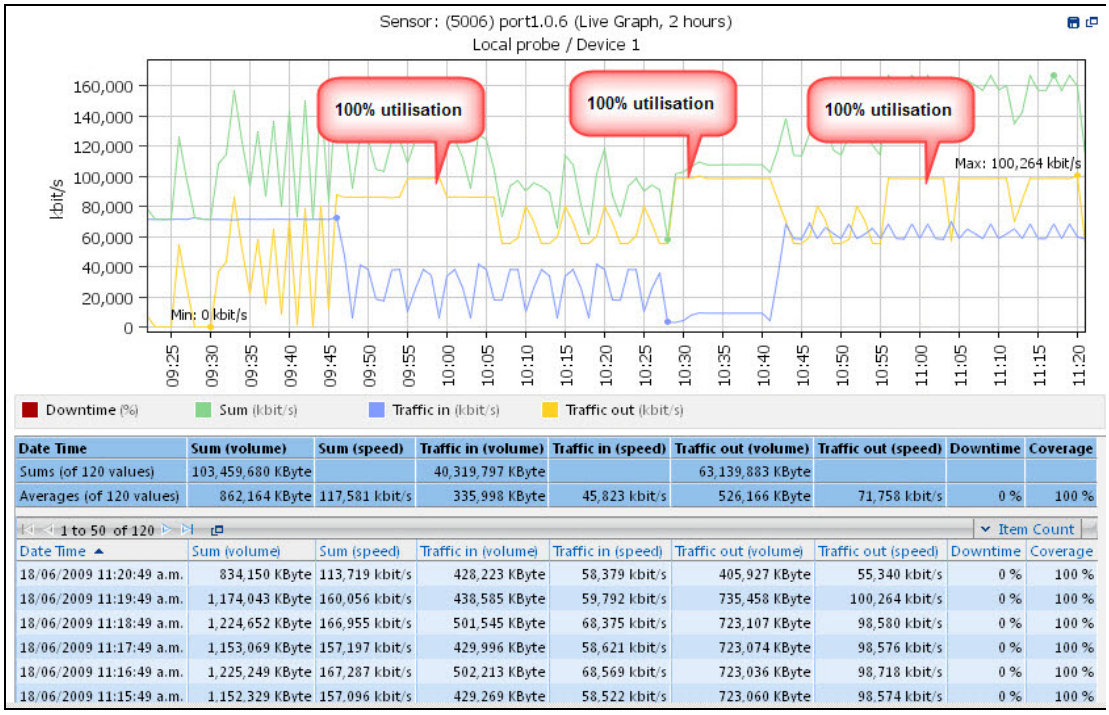
An oversubscribed port is one from which the switch is attempting to deliver data at a rate higher than the port's bandwidth can allow.



When this occurs, some packets must end up being dropped. The users whose packets are being dropped will experience a slow-down in their file transfers or in the responsiveness of the network-based applications they are running.

An important step in diagnosing network slowdowns is to look for oversubscribed ports in the network. One way to find oversubscribed ports is to use a tool that will report the utilisation of network ports. A variety of such tools exist, most network management packages provide facilities for monitoring port utilisation. The popular traffic graphing tool MRTG is widely used for the task of port utilisation monitoring and graphing. Other commercial tools, similar to MRTG, e.g. PRTG, provide similar port utilisation monitoring and graphing capabilities.

Typically these tools will provide graphs of utilisation over time, similar to the one below:



Date Time	Sum (volume)	Sum (speed)	Traffic in (volume)	Traffic in (speed)	Traffic out (volume)	Traffic out (speed)	Downtime	Coverage
Sums (of 120 values)	103,459,680 KByte		40,319,797 KByte		63,139,883 KByte			
Averages (of 120 values)	862,164 KByte	117,581 kbit/s	335,998 KByte	45,823 kbit/s	526,166 KByte	71,758 kbit/s	0 %	100 %

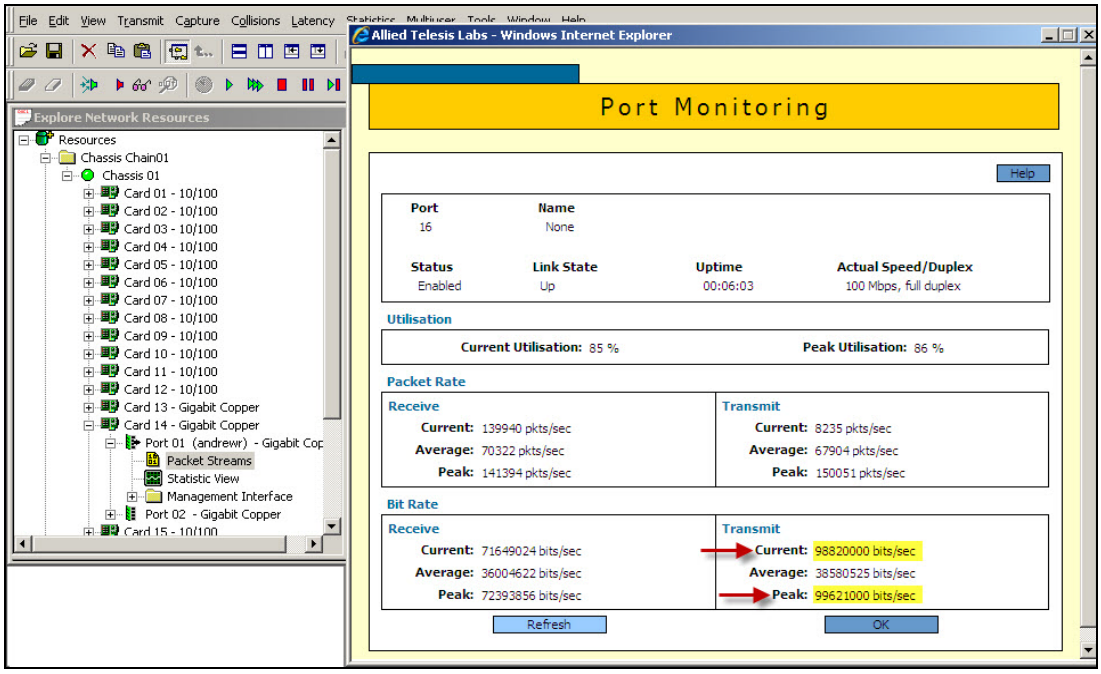
1 to 50 of 120 — Item Count

Date Time	Sum (volume)	Sum (speed)	Traffic in (volume)	Traffic in (speed)	Traffic out (volume)	Traffic out (speed)	Downtime	Coverage
18/06/2009 11:20:49 a.m.	834,150 KByte	113,719 kbit/s	428,223 KByte	58,379 kbit/s	405,927 KByte	55,340 kbit/s	0 %	100 %
18/06/2009 11:19:49 a.m.	1,174,043 KByte	160,056 kbit/s	438,585 KByte	59,792 kbit/s	735,458 KByte	100,264 kbit/s	0 %	100 %
18/06/2009 11:18:49 a.m.	1,224,652 KByte	166,955 kbit/s	501,545 KByte	68,375 kbit/s	723,107 KByte	98,580 kbit/s	0 %	100 %
18/06/2009 11:17:49 a.m.	1,153,069 KByte	157,197 kbit/s	429,996 KByte	58,621 kbit/s	723,074 KByte	98,576 kbit/s	0 %	100 %
18/06/2009 11:16:49 a.m.	1,225,249 KByte	167,287 kbit/s	502,213 KByte	68,569 kbit/s	723,036 KByte	98,718 kbit/s	0 %	100 %
18/06/2009 11:15:49 a.m.	1,152,329 KByte	157,096 kbit/s	429,269 KByte	58,522 kbit/s	723,060 KByte	98,574 kbit/s	0 %	100 %

Periods during which the port is at **100% utilisation** are sure signs of the port being oversubscribed.

Using the GUI to monitor port data transmission

Port data transmission rates can also be monitored through the switch's GUI interfaces. If the transmit rate is frequently close to the port's maximum bandwidth, then it is highly likely that congestion is occurring on the port. Note the highlighted **Current** and **Peak** values shown for port 16 in the diagram below.



Checking the state of egress port queues

Another effective way to find ports that are oversubscribed is to look at the state of each switch port's egress queues. If there are packets queued up in a port's egress queues, then the port is oversubscribed.

On x900 and x908 switches, in order to see the state of port egress queues, it is necessary to enable Quality of Service (QoS) counters, using the command:

```
platform enhancedmode {nexthop|qoscounters}
```

followed by a reboot of the switch.

Then, after the switch has rebooted, the egress queue counters can be viewed with the following command:

```
show mls qos interface <interface name> queue-counters
```

This will output a snapshot of the state of the egress queues on the specified port. If some queues regularly have non-zero values for a given port, then that port is oversubscribed.

Data Corruption

If packets are being corrupted, by faulty cabling, or by electrical interference, or by switch hardware faults, then the corrupted packets will be dropped by the receiving switch. A high rate of corrupt packets will cause a slow-down in network performance, as the hosts sending the dropped packets will have to resend them.

To look for evidence of a port receiving corrupt packets, use the command:

```
show platform table port <interface name> counters
```

For example:

```
show platform table port port1.0.18 counters
```

Look for non-zero error counters in the **Receive** column.

If you find a port that is receiving a large rate of errors, as shown in bold in the output below, then that will indicate that either the cable connected to the port is faulty, or the port on the other end of the cable is faulty.

```
 Triple-Auth#sh plat table port port1.0.13 count

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

Port port1.0.13 Ethernet MAC counters:
 Combined receive/transmit packets by size (octets) counters:
  64                          29293 1024 - MaxPktSz               4713
  65 - 127                   434884 1519 - 1522                     0
  128 - 255                   66732 1519 - 2047                     0
  256 - 511                    2282 2048 - 4095                     0
  512 - 1023                    248 4096 - 9216                     0

 General Counters:
 Receive                            Transmit
  Octets                   35868103 Octets                   25871611
  Pkts                       323030 Pkts                       215122
  FCSErrors                       0
  MulticastPkts               48940 MulticastPkts                   4
  BroadcastPkts               21054 BroadcastPkts                1309
  PauseMACCtlFrms                 0 PauseMACCtlFrms                0
  OversizePkts                    0
  Fragments                       3
  Jabbers                        21
  UnsupportOpcode                 0
  AlignmentErrors               193
  SymErDurCarrier                 0
  CarrierSenseErr                 0
  UndersizePkts                 743
                                     FrameWDeferrdTx               0
                                     FrmWExcesDefer                0
                                     SingleCollsnFrm               0
                                     MultCollsnFrm                 0
                                     LateCollisions                0
                                     ExcessivCollsns               0
                                     Collisions                    0

 Layer 3 Counters:
  ifInUcastPkts               253036 ifOutUcastPkts              213809
  ifInDiscards                 39423 ifOutErrors                     0
  ipInHdrErrors                    6

 Miscellaneous Counters:
  DropEvents                      0
  ifOutDiscards                   0
  MTUExcdDiscard                  0

------------------------------------------------------------------------
```

Collisions

One special case to consider is when a port is reporting a high rate of collisions. On full-duplex ports, there should **never** be any collisions, as sent and received packets are exchanged in different channels, and cannot collide with each other.

If a port is reporting a high rate of collisions, it is highly likely that the port is in half-duplex mode, and the port on the switch at the other end of the link is in full-duplex mode.

As a result, the switch in full-duplex mode will send data even while the half-duplex port is transmitting (which is normal behaviour for a full-duplex port). The half-duplex port, though, will see this situation as causing collisions, and will frequently abort packet sending, and retry. This can significantly reduce throughput. This duplex mismatch commonly occurs if one end of the link has a fixed speed/duplex, and the port at the other end is autonegotiating. The autonegotiating port will not be able to detect the duplex state of the peer port, and will default to half duplex.

The solution to the duplex mismatch problem is to **change the configuration of the ports** to ensure that they will come up with the same duplex state at both ends.

Checking for collisions

Use the following command to check for collisions:

```
show platform table port <port-number> count
```

The command will output a table of counters for a port, among the counters are the number of collisions that have occurred on the port - they are shown in bold in the example below:

```
awplus#show platform table port port1.0.13 count
Switch Port Counters
------------------------------------------------------------------------

Port port1.0.13 Ethernet MAC counters:
 Combined receive/transmit packets by size (octets) counters:
  64                            1084 1024 - MaxPktSz               7040
  65 - 127                      2773 1519 - 1522                      0
  128 - 255                      162 1519 - 2047                      0
  256 - 511                       10 2048 - 4095                      0
  512 - 1023                     106 4096 - 9216                      0
 General Counters:
 Receive                             Transmit
  Octets                   10820753 Octets                     220353
  Pkts                         9484 Pkts                         1691
  UndersizePkts                   0

 <Some output removed for brevity>
  FrameWDeferrdTx                 2
                                      FrmWExcesDefer                0
                                      SingleCollsnFrm               72943
                                      MultCollsnFrm                 24527
                                      LateCollisions                2694
                                      ExcessivCollsns               571
                                      Collisions                    951313
 Layer 3 Counters:
  ifInUcastPkts                7254 ifOutUcastPkts                490
  ifInDiscards                 2147 ifOutErrors                     0
  ipInHdrErrors                   0
```

The definitions of the collision-related counters output by this command are:

Collisions - This is a counter of all collisions that have occurred on that port. A collision being a case of a half-duplex interface detecting an incoming packet at the time it was trying to transmit a packet.

Single Collision Frames, Multiple Collision Frames - These counters indicate how many times the port has experienced a single collision when attempting to transmit a given frame, and how many times it experienced multiple collisions whilst attempting to deliver a given frame.

Late Collision - A late collision occurs when the switch detects an incoming packet after it has already transmitted 64 bytes of its current outgoing packet. In a properly constructed Ethernet network this should never happen, as other hosts on the segment should not start transmitting when one host has already been transmitting for 64-bytes worth of time. However, in cases of a duplex mismatch, late collisions are highly likely, as a port at one end of the link is operating at full duplex. The full-duplex port has no problem with transmitting at the same time as the port at the other end of the link is transmitting, as it considers each switch's transmissions to be in separate logical channels, and therefore unable to collide. The port operating in half-duplex, however, considers both switches to be transmitting in the same logical channel. So if it is well advanced in transmitting a packet when the peer switch starts a transmission, it will experience a late collision.

Excessive Collisions - Each time a port experiences a collision when attempting to transmit, it will pause, and then try again. If the port has 16 attempts to transmit a packet and each results in a collision, then the port increments the ExcessiveCollisions counter, and gives up attempting to deliver the packet. Again, in a case of duplex mismatch, if the port operating in full duplex is transmitting at a high data rate, it is possible for the port operating in half duplex to experience cases of excessive collisions.

A high rate of STP Topology Change Notifications

There is another relatively common cause of intermittent performance problems in networks that are using spanning tree (any type of spanning tree - STP, RSTP or MSTP) to control redundant network paths. When an active STP port changes link status, the switch will send a Topology Change Notification (TCN) to the other switches in the spanning tree. When those other switches receive the TCNs, they need to flush all or part of their MAC and ARP tables.

Then, of course, those entries need to be relearnt. During the relearning period, all switches in the network will flood almost all the packets they receive, as though they were hubs. This will cause a period of congestion, and quite possibly high CPU utilisation on workstations and servers, as they experience a significantly increased rate of packet reception (most of which are not actually intended for them). This all causes a brief slow down of network performance.

If the topology changes keep happening on the network with some frequency for an extended period, then the slow down of network performance will become quite noticeable.

This problem of frequent topology changes over an extended period is often caused by edge ports on the network not being configured as **portfast**. Edge ports are those ports that connect to workstations or servers, as against those ports that connect to other switches. In normal

circumstances, these ports are effectively 'leaves' on the spanning tree, as they connect to end-point devices. Because of this, the spanning tree does not need to be informed about state changes of these ports, as those state changes do not result in changes to the internal topology of the spanning tree.

When portfast is configured on an edge port, then the turning on or off, or rebooting, of the PC connected to the port will not result in TCNs. However, a failure to configure portfast on the edge ports of the network will result in TCNs being generated, and propagated through the whole spanning tree, every time those ports change state. Consequently, at times of the day when many PCs are being turned on or off there will be a noticeable slowing of the network.

Therefore, one avenue of investigation of network slow downs (particularly slow downs at the start and end of the working day) is to check whether the edge ports on the network have been configured for **portfast**.

Determining where the topology changes are coming from

If you have a number of access switches in the network, and are confident that almost all of them have been correctly configured with portfast on their workstation-connected ports, it could be a tedious job to go around all the access switches looking for the one or two that are not yet configured correctly.

Fortunately, the AW+ software can help you identify the location of the switch(es) generating the topology changes.

If you enable STP topology-change debugging, with the command **debug mstp topology-change**, then the receipt of a topology change notification will result in the output of a message identifying the port on which the notification was received.

That, of course, will direct you towards the source of the topology changes, and therefore, towards the access switch that is not yet correctly configured.

Other points to note in relation to network slowdowns

High CPU utilisation on switches does not cause network slowdowns

It is important to keep in mind that Ethernet switches forward packets in their switching ASICs. Although the CPU is involved in processing network control protocols, and populating the ASIC forwarding tables, the **actual packet forwarding is performed by the ASICs**.

In fact, the rate at which a switch forwards packets is quite unrelated to how busy its CPU is.

Despite that, it is not uncommon for a switch's CPU to experience high utilisation as a consequence of something that is causing network congestion. In particular, if there is a high level of broadcast packets flowing in the network, that could result in network congestion, and also result in high utilisation of switches' CPUs, as broadcast packets are all sent to the CPU, as well as forwarded.

Whilst switches with high CPU utilisation are not a cause of a network slowdown, the occurrence of high CPU utilisation on the switches at the same time as a network slowdown is well worth investigating, as it could provide important clues as to the actual cause of the slowdown.

In particular, if the high CPU utilisation is (as is most likely) being caused by a high rate of data coming up to the switches' CPUs, then identifying this CPU-directed data could help in identifying what data is causing congestion in the network.

Identifying data causing congestion in the network

Allied Telesis x-series switches provide a variety of methods and commands for obtaining a capture of the packets arriving at the CPU. Here are two useful commands:

Packet snapshot

To get a snapshot of the packets coming up to the CPU, use the command:

```
debug platform packet timeout 2
```

This will give you a 2 second snapshot of all the packets coming up to the CPU. Note that it might take well more than 2 seconds, quite possibly some minutes, for the debug output to complete.

Identify traffic type

The next step is to investigate this debug capture to determine what the bulk of the traffic is, and where it is coming from. If the bulk of the packets in the capture are IP packets or ARP, then you can obtain a more decoded, and finer-grained packet trace by using the command:

```
debug ip packet interface {<interface-name>|all}[address
  <ipaddress>|verbose|hex|arp|udp|tcp|icmp]
```

This command operates in a manner that is very similar to the popular **tcpdump** utility that is available on numerous computing platforms. With the help of this command, you can obtain a lot of valuable troubleshooting information:

- You can get a textual decoding of the packets.
- By specifying different interfaces in the command, you can narrow down the ingress interface of the heavy packet streams.
- By specifying protocols (ARP, TCP, UDP, ICMP), you can quickly narrow down on the traffic type.

Having identified the traffic type (IP or other) that is over-utilising the CPU, the next step is to work out where it is coming from, and this may lead you to whatever network problem that is causing that unnecessary traffic to be sent to the CPU of the switch, and probably also causing network congestion.

Ping can be a useful tool in monitoring for network slowdown

Ping is a simple, effective, way to monitor the round-trip time along particular network paths. By pinging between various pairs of points in the network, it is often possible to identify the point(s) in the network at which packets are being queued or dropped.

But, it is important to have a clear understanding of the best way to make use of ping.

Simply looking at the average round-trip-time of the pings between one device and another does not necessarily provide very useful information about network performance. Typically, the bulk of the round-trip-time of a ping is not due to switch forwarding latency. Rather the bulk of the time is due to the software processing of the ping packet in the sending and receiving devices. **Different networking devices process ping packets differently** - some will treat ping packets as high priority and respond to them very quickly; others will not give such high priority to ping, which will result in a longer round-trip-time.

Consistently high round-trip time for pings to an Ethernet switch does not provide any evidence that the switch is experiencing congestion, or even that its CPU is heavily loaded, it just indicates that this switch does not have a fast turn-around of ping packets.

When using ping to help investigate network slow-downs, it is far more important to look for significant:

- **variations** in ping round-trip times along a given network path
- numbers of pings on a given network path getting **no response**

A significant variation of ping round-trip times, or intermittent loss of ping responses, could be an indicator of network congestion.

Packets sitting in egress queues when ports are oversubscribed can cause variation in ping round-trip times. Similarly, when oversubscribed ports have to drop packets, that can cause intermittent loss of ping responses.

However, be aware that significant variation of ping round-trip times, or intermittent loss of ping responses are not highly reliable indicators of network congestion. These variations could just as easily be caused by variable CPU loading of the target device.

So, while ping is useful as a simple way to monitor whether there is variation in the loading on the network or the network's hosts, it is too simple to be a useful diagnostic tool.

Once you have reason to believe that there are over-loaded or slow-performing paths in your network, then further investigation needs to gather more precise information than can be gained from ping. You need to move on to looking at port utilisations, errored packet counters, etc., as have been described in this How To Note.

C613-16145-00 REV B

Allied Telesis™ the solution : the network

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

alliedtelesis.com