AGENDA

- Know Your Network
- Action Plans / Problem Determination
- Tools - General Usage
  (ping, traceroute, netstat, nslookup, ...)
- Understanding the Common Tools
- Problem Diagnosis Tips
Know Your Network! . . .

- In order to manage any network successfully, you must be aware of the topology.

- Before any successful, and timely, problem resolution can be attempted, a (current !) network diagram is *essential*.

- The diagram (and associated documentation) should indicate all nodes and all possible paths, and detail the subnets, addresses and software (especially versions) available at each node.

- *Only then is it possible to create an appropriate action plan...*
• **Where to Start?** - First, *identify the problem*. This will determine the right tools to use, and the right place to start testing from (! “Top-down” or “Bottom-up” !). Progressive testing may be needed to isolate the problem area.

**Misinformation Anecdote**

• Network problems usually fall into two or three categories:-
  • *No connection can be made.*
  • *Connections can be made, but are unstable, OR , not all functions operate.*
  • *Connections are stable but performance is poor.*
Action Plans . . . . .

**Connectivity** issues can be caused by:-

- Application errors
- Failed network connections
- Bad configuration/changes
- Hardware failures
- Failed bind
- Power failures
- Security restrictions

**Performance** issues can be caused by:-

- Insufficient bandwidth
- Bottlenecks
- Priorities
- Retries
- Broadcasts
- Congestion
- Routing
- Fragmentation
- Application errors
- Switch faults
1. **Investigate (ALL) error messages** – these may indicate the nature and location of the failure [e.g. “ttl” expired, no path available, packet size too large (“nofragment” is on)].

2. **Classify the error** – ask what works and what doesn’t, and for whom . . .

- Problems affecting one person are likely to be physical (e.g. check cables/switch/vlan first).
- Problems affecting more than one user are more likely to be the network.
- Problems affecting more than one person & more than one network path are more likely to be the application.
3. **Test connectivity** *(end-to-end)* – using Ping/Traceroute. Be careful to ensure that the packets take the same path as the problem connection (i.e. ensure correct source interface address – you may need to use an “extended” PING).

- If PING fails, note the location and investigate there.

- If PING succeeds (note that this is ICMP, the connection probably uses TCP, so this may NOT be a conclusive test), try with a TCP PING if available.

- If PING succeeds try again with larger packets, if appropriate.
For Example: Problem reported as ...
“end-user cannot connect to application”

- Starting at the end-user system ensure local physical connections are good, then check the next layer, such as local switch ports, vlans, routers, and even firewalls.
- Then, test each “hop” by progressive steps across the network.
- Then ensure that the system running the required application is connected at the network level (“ping” from that system outbound via the interface in question.

If all these results are good, then the issue is probably with the application and not a network problem!
Disclaimer:

The fact that some tools are mentioned in this presentation while other tools are not, in no way implies recommendation of the tools mentioned, nor condemnation of those tools not mentioned.

The purpose of this presentation is simply to make attendees aware that such tools exist, and the attendees should make up their own mind as to the suitability of any tool used on their own system.
“Common” Tools . . . . .

“PING” - proves that connectivity exists

“TRACERTE” - discovers the network path (also “tracert”)

“NETSTAT” - to locate connection information

| ALL       | - All connections to a stack          |
| ALLConn   | - TCP/IP connections                  |
| ARp       | - Query ARP table or entry information|
| CONFIG    | - Configuration data                  |
| Conn      | - Active TCP/IP connections (Default) |
| Devlinks  | - Devices and links                   |
| Gate      | - Current known gateways              |
| HOME      | - Home address list                   |
| PORTList  | - Display port reservation list        |
| ROUTe     | - Display routing information         |
| SOCKets   | - Socket interface users and sockets   |
| STATS     | - TCP/IP statistics                   |
| TCP       | - Displays detailed info about the stack|
| TELnet    | - Telnet connection information       |

z/OS command format:

```
-------------------- -------------------- -------------------- --------------------
| NETSTAT < Option | Command > < NETSTAT < Option | Command > < NETSTAT < Option | Command > < NETSTAT < Option | Command > <
| Target > 
| Output > < (Select >
```

E.g.:
```
TSO NETSTAT CONN (PORT 25
TSO NETSTAT TCP TCPIP
```

Note that “NETSTAT . . . . (REPORT” will collect the output to a dataset; for ease of reading or input to a REXX.
“Nslookup” - test domain name resolution (& “DIG”)

“Snmp” - where SNMP is supported, there are many tools available to extract further information (MIB data), once the problem area has been located (e.g. Monitors, such as “Implex” for z/OS; “iReasoning” elsewhere)

“TIVOLI” - IBM network tools (Monitor and trace facilities)

“Ctrace” - z/OS trace tool
“EXIGENCE” - WDS trace “expert” system
(now ZTS! – “ZEN Trace & Solve“)
“TPing” - (“TurboPing”) “PING” using TCP packets

“Tcpdump” - (also Windump & SSLdump) is a packet sniffer found on many (most?) open platforms.

“Ethereal” - open system packet analyser (& “Wireshark”)

“Pchar” - is a reimplementation of Van Jacobson's (“Mr Traceroute”) pathchar utility which analyses the individual hops of a path.

“Netcat” - Netcat is a utility which reads and writes data across network connections. It is a network debugging and exploration tool. (+ port-scanner !)

“VisualRoute” - path checker and graphical display

“NeoTrace” - (McAfee) Internet locator: enhanced traceroute

....etc
"Ping" - "Packet INternetwork Groper", is usually ICMP-based, which works if ICMP is allowed to pass. If not permitted, then an application-based ping can be used [e.g. "APING" (UDP) or "TPing" (TCP)].

Ping tests by sending out ICMP Request packets, and receiving ICMP Replies, therefore verifying up to (ISO) layer 3 . . .

C:\>ping 66.249.85.99           ( www.google.co.uk       ----- use IP address or URL )
Pinging 66.249.85.99 with 32 bytes of data:

Reply from 66.249.85.99: bytes=32 time=22ms TTL=244
Reply from 66.249.85.99: bytes=32 time=22ms TTL=244
Reply from 66.249.85.99: bytes=32 time=42ms TTL=244
Reply from 66.249.85.99: bytes=32 time=22ms TTL=244

Ping statistics for 66.249.85.99: Packets: Sent=4, Recvd=4, Lost=0 (0% loss), Approx. round trip times in milliseconds: Min=22ms, Max=42ms, Ave=27ms
ISO 7-Layer Network Model

Layer 1: Physical - defines the real hardware.
Layer 2: Data Link - defines the format of data (frame/packet). (MAC)
Layer 3: Network - responsible for routing datagrams. (IP)
Layer 4: Transport - manages data between network and user. TCP/UDP)
Layer 5: Session - defines the format of the data sent.
Layer 6: Presentation - converts to/from local representation of data.
Layer 7: Application - provides network services to the end-users.

TCP/IP 4-Layer (Unix/DoD) Network Model

Layer 1: Link - defines the network hardware and device drivers.
Layer 2: Network - addressing, routing, delivery. (IP / ICMP) (ARP)
Layer 3: Transport - communication; end-to-end integrity. (TCP / UDP)
Layer 4: Application - user applications.
   (DNS, arp, telnet, smtp, http, ftp, traceroute....)
ICMP Types:

0  Echo Reply
3  Destination Unreachable
4  Source Quench
5  Redirect
6  Alternate Host Address
8  Echo
9  Router Advertisement
10 Router Solicitation
11 Time Exceeded
12 Parameter Problem
13 Timestamp
14 Timestamp Reply
15 Information Request
16 Information Reply
17 Address Mask Request
18 Address Mask Reply
30 Traceroute
31 Datagram Conversion Error
32 Mobile Host Redirect
33 IPv6 Where-Are-You
34 IPv6 I-Am-Here
35 Mobile Registration Request
36 Mobile Registration Reply
37 Domain Name Request
38 Domain Name Reply

ICMP Codes:

3  Destination Unreachable
0  Net Unreachable
1  Host Unreachable
2  Protocol Unreachable
3  Port Unreachable
4  Fragmentation Needed and DF Set
5  Source Route Failed
6  Destination Network Unknown
7  Destination Host Unknown
8  Source Host Isolated
9  Communication with Dest Network Prohibited
10 Communication with Dest Host Prohibited
11 Dest Network Unreachable for Type of Service
12 Dest Host Unreachable for Type of Service
13 Communication Administratively Prohibited
14 Host Precedence Violation
15 Precedence cutoff in effect

11 Time Exceeded
0  Time to Live exceeded in Transit
1  Fragment Reassembly Time Exceeded

Ref: “www.iana.org/assignments/icmp-parameters”
PING (Windows)

         [-r count] [-s count] [[-j host-list] | [-k host-list]]
         [-w timeout] target_name

Options:
  -t  Ping the specified host until stopped.
      To see statistics and continue - type Control-Break;
      To stop - type Control-C.
  -a  Resolve addresses to hostnames.
  -n  count  Number of echo requests to send.
  -l  size   Send buffer size.
  -f  Set Don't Fragment flag in packet.
  -I  TTL   Time To Live.
  -v  TOS   Type Of Service.
  -r  count Record route for count hops.
  -s  count Timestamp for count hops.
  -j  host-list  Loose source route along host-list.
  -k  host-list  Strict source route along host-list.
  -w  timeout Timeout in milliseconds to wait for each reply.
PING

C:\>ping 66.249.85.55  ← non-existent addresses
Pinging 66.249.85.55 with 32 bytes of data:

Request timed out.
Request timed out. (or "Destination Unreachable ?)
Request timed out. (if a return path is available)
Request timed out.

Ping statistics for 66.249.85.55: Packets: Sent=4, Recvd=0, Lost=4 (100% loss),

Drawbacks:

- Extra traffic on the network.
- "Time To Live" (TTL) set to a high value to ensure penetration.
- Network devices **may not allow** Ping/ICMP and may drop its priority.
- May not take the same path as user traffic; delay (latency) reported may **not** be representative for the application(s).
- Low feedback on fault and location.
**TRACEROUTE** (Windows)

**Usage:** `tracert [-d] [-h maximum_hops] [-j host-list] [-w timeout] target_name`

**Options:**
- `-d`  Do not resolve addresses to hostnames.
- `-h maximum_hops`  Maximum number of hops to search for target.
- `-j host-list`  Loose source route along host-list.
- `-w timeout`  Wait timeout milliseconds for each reply.

- Also uses ICMP ! (although some platforms use UDP)
- Good for spotting “loops” in the routing
- **“Time To Live” (TTL*)** is incremented for each positive response.
- Each “hop” in the path is identified (Names may be resolved!).
- “Per hop” round-trip delays can be identified.
- **Drawbacks** are similar to those of “Ping”.

( * = anti-looping function of TCP/IP )
Tools in Detail . . . . .

TRACEROUTE

C:\>tracert 66.249.85.55 ( www.google.co.uk ----- use IP address or URL )

Tracing route to 66.249.85.55 over a maximum of 30 hops

1  1 ms  1 ms  1 ms  81.144.212.33
2  7 ms  6 ms  6 ms  62.7.96.41
3  6 ms  6 ms  6 ms  core2-gig2-1.kingston.ukcore.bt.net [194.72.3.2]
4  7 ms  7 ms  7 ms  core2-pos7-3.ealing.ukcore.bt.net [62.6.201.42]
5  7 ms  7 ms  7 ms  core2-pos10-0.redbus.ukcore.bt.net [194.74.65.202]
6  8 ms  7 ms  8 ms  194.74.65.38
7  7 ms  7 ms  7 ms  72.14.238.244
8 16 ms 16 ms 16 ms  216.239.43.91
9 22 ms 22 ms 22 ms  72.14.232.209
10 *     *     *     Request timed out.
11 *     *     *     Request timed out.
12 *     etc,etc . . .  <----- default maximum of 30

TRACEROUTE should be run in BOTH directions!!

Look for unsuitable (long) routes and high latency
Some platforms give status indicators...

!H - Host unreachable. (Destination Net unreachable) The router has no route to the target system.

!N - Network unreachable.

!P - Protocol unreachable.

!S - Source route failed. A router is blocking source-routed packets.

!F - Fragmentation needed. (Check the MTU configuration at the router).

!X - Communication administratively prohibited. Traceroute blocked!

TRACEROUTE can be enhanced by visualization, as is often seen in graphical traceroute tools: such as . . .
TraceRoute Tools . . . .

VisualRoute - 1

Report for www.google.co.uk [66.249.85.99]

Analysis

This trace was started on 09-Jan-2007 10:29:48. The host 'www.google.co.uk' (known as ff-in-f99.google.com) has been found, and is reachable in 10 hops. Also, it responded to HTTP requests on port 80 (it is running server GDS/2.1, which responded in 431ms). The TTL value of packets received from it is 246.

In general, this route offers a good throughput, with hops responding on average within 11ms. The DNS lookup was completed almost instantaneously (less than 2ms - this may be the result of caching).
TraceRoute Tools .......

VisualRoute - 2

Learn more at:
http://www.visualroute.com
### TraceRoute Tools

#### PingPlotter

**Target Name:** www.google.co.uk  
**IP:** 66.249.85.99

<table>
<thead>
<tr>
<th>Hop</th>
<th>PL%</th>
<th>IP</th>
<th>DNSName</th>
<th>Avg</th>
<th>Cur</th>
<th>Graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>81.144.212.33</td>
<td>---------------</td>
<td>----------------------------</td>
<td>1</td>
<td>1</td>
<td>23</td>
</tr>
<tr>
<td>2</td>
<td>62.7.96.41</td>
<td>---------------</td>
<td>----------------------------</td>
<td>6</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>194.72.3.2</td>
<td>core2-gig2-1.kingston.ukcore.bt.net</td>
<td>6</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>62.6.201.42</td>
<td>core2-pos7-3.ealing.ukcore.bt.net</td>
<td>7</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>194.74.65.202</td>
<td>core2-pos10-0.redbus.ukcore.bt.net</td>
<td>7</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>194.74.65.38</td>
<td>---------------</td>
<td>----------------------------</td>
<td>7</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>72.14.238.244</td>
<td>---------------</td>
<td>----------------------------</td>
<td>7</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>216.239.43.91</td>
<td>---------------</td>
<td>----------------------------</td>
<td>16</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>72.14.232.209</td>
<td>---------------</td>
<td>----------------------------</td>
<td>21</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>66.249.85.99</td>
<td>ff-in-f99.google.com</td>
<td>21</td>
<td>22</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Round Trip:** 21 22

Data and Image generated by Ping Plotter Freeware (http://www.pingplotter.com)
Where the target system is external to the local network, and especially where routing is not available to/from the local network, there are several sites around the World that offer the ability to run “Ping” and “Traceroute” to be instigated by remote control from their web site.

Basically, this is a “proxy” service; the remote site issuing the test on your behalf.

This is suitable for determining the general availability of the target system (i.e. from anywhere on the Internet), but does not test specific routes.

“www.samspade.org” used to be an excellent example of this type of service, but is not currently available in its previous form.

Further directions to such services can be found at:

“wwwtraceroute.org”
NETSTAT(z/OS)

NETSTAT < Option | Command > < Target >
    < Output > < (Select >

TSO NETSTAT CONN    TSO NETSTAT SOCK
TSO NETSTAT DEV     TSO NETSTAT ROUTE
TSO NETSTAT TCP TCPPIP

Also “onetstat”...

Can be issued from either TSO or USS; the results are the same.

NB. Netstat options will vary depending upon the platform!

Note the following examples from z/OS and Windows. . .
### Tools in Detail

#### NETSTAT(z/OS) – "DEV"

<table>
<thead>
<tr>
<th>DevName</th>
<th>DevType</th>
<th>DevNum</th>
<th>Status</th>
<th>LnkName</th>
<th>LnkType</th>
<th>LnkStatus</th>
<th>NetNum</th>
<th>QueSize</th>
<th>IpBroadcast</th>
<th>MacAddress</th>
<th>ActMtu</th>
<th>BSD Routing Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCS1</td>
<td>LCS</td>
<td>0E20</td>
<td>Ready</td>
<td>ETH1</td>
<td>ETH</td>
<td>Ready</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>000255305115</td>
<td>1500</td>
<td></td>
</tr>
</tbody>
</table>

#### NETSTAT(z/OS) – "SOCK"

<table>
<thead>
<tr>
<th>Name</th>
<th>Subtask</th>
<th>Type</th>
<th>Status</th>
<th>Conn</th>
<th>BoundTo</th>
<th>ConnTo</th>
<th>Group</th>
<th>RefCnt</th>
<th>BytesIn</th>
<th>Inbound Packets</th>
<th>Inbound Packets In Error</th>
<th>Inbound Packets Discarded</th>
<th>Inbound Packets With No Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>APIASHB</td>
<td>007E1048</td>
<td>Dgram</td>
<td>UDP</td>
<td>00001A1A</td>
<td>192.168.1.156..12004</td>
<td>00001A1A</td>
<td>224.0.0.1</td>
<td>00000000</td>
<td>420328206</td>
<td>2865741</td>
<td>1360</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>APIASHB</td>
<td>007E12D8</td>
<td>Stream</td>
<td>Listen</td>
<td>00001A19</td>
<td>192.168.1.156..12004</td>
<td>00001A19</td>
<td>224.0.0.1</td>
<td>00000000</td>
<td>420328206</td>
<td>2865741</td>
<td>1360</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

- **-a** Displays all connections and listening ports.
- **-n** Displays addresses and port numbers in numerical form.
- **-r** Displays the routing table.

...etc

C:\>netstat -a

**Active Connections**

<table>
<thead>
<tr>
<th>Proto</th>
<th>Local Address</th>
<th>Foreign Address</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>wdsdgw:epmap</td>
<td>0.0.0.0:0</td>
<td>LISTENING</td>
</tr>
<tr>
<td>TCP</td>
<td>wdsdgw:microsoft-ds</td>
<td>0.0.0.0:0</td>
<td>LISTENING</td>
</tr>
<tr>
<td>TCP</td>
<td>wdsdgw:1028</td>
<td>0.0.0.0:0</td>
<td>LISTENING</td>
</tr>
<tr>
<td>TCP</td>
<td>wdsdgw:1241</td>
<td>0.0.0.0:0</td>
<td>LISTENING</td>
</tr>
<tr>
<td>TCP</td>
<td>wdsdgw:10110</td>
<td>0.0.0.0:0</td>
<td>LISTENING</td>
</tr>
<tr>
<td>UDP</td>
<td>wdsdgw:microsoft-ds</td>
<td><em>:</em></td>
<td></td>
</tr>
<tr>
<td>UDP</td>
<td>wdsdgw:isakmp</td>
<td><em>:</em></td>
<td></td>
</tr>
<tr>
<td>UDP</td>
<td>wdsdgw:1033</td>
<td><em>:</em></td>
<td></td>
</tr>
<tr>
<td>UDP</td>
<td>wdsdgw:4500</td>
<td><em>:</em></td>
<td></td>
</tr>
<tr>
<td>UDP</td>
<td>wdsdgw:ntp</td>
<td><em>:</em></td>
<td></td>
</tr>
<tr>
<td>UDP</td>
<td>wdsdgw:1900</td>
<td><em>:</em></td>
<td></td>
</tr>
</tbody>
</table>
In general, it is quite common to seek an IP target using a URL (which acts rather like a PATH name).

This entails sending the URL to a “Domain Name Server” (or “Resolver”) in z/OS terms) to have the name translated (i.e. a “table lookup”) into an IP address (this may occur locally by use of the “Hosts” file).

The IP address returned is then used to address the target.

This process may also be performed in reverse; i.e. the DNS server can translate an IP address into a URL!

The use of a URL means that remote services can be failed-over, relocated or rebuilt without the users needing to know!
The global Domain Name System is a hierarchy of servers/services spread across the Internet. At its core is a set of servers that manage the base domains; such as “com”, “edu”, “gov” ...etc

When a name is “looked up” it happens from right to left - recursively.

Take www.google.co.uk ...
  - First the server is located that controls the “uk” domain (there is an implied “root” service where all top-level servers are known).
  - This will indicate the “co.uk” server; which in turn will indicate the “google.co.uk” server.
  - The “google.co.uk” server will have IP addresses (an “A” record) for web (“www”) and mail services (note: “www” is not the only canonical form used!)

NAMED.CONF - lists the “zones” (eg. “google.co.uk”)
ZONE FILES - hold the IP addresses

NB. Zone information changed at the bottom of a “layer” is propagated upwards by “Zone Transfer” at preset times.
### NSLOOKUP (Windows)

#### Usage:
```
nslookup NAME  , or , NAME1 NAME2
```

or
```
command
```

#### set option
```
all [no]debug [no]d2 [no]defname
srchlist=N1[/N2/.../N6] root=NAME retry=x
timeout=X type=X querytype=X class=X
[no]msxfr ixfrver=X
```

#### Server NAME

#### Exit

---

“Lookup” failure will cause connectivity failure, and symptoms can be mistaken for a routing problem!

---

z/OS often acts as a relay, passing the requests on to a network DNS server.
NSLOOKUP (Windows)

C:\>nslookup

> set debug
> www.google.co.uk
Server: my.router
Address: 192.168.27.1
---------- (debug information)
Got answer:

HEADER:
  opcode = QUERY, id = 3, rcode = NOERROR
  header flags: response, want recursion, recursion avail.
  questions = 1, answers = 1, authority records = 0, additional = 0

QUESTIONS:
  www.google.co.uk.uk.willdata.com, type = A, class = IN

ANSWERS:
  -> www.google.co.uk.uk.willdata.com
    internet address = 212.69.199.183
    ttl = 60 (1 min)

----------
Non-authoritative answer: ←----- (Retrieved from a cache! )
Name:   www.google.co.uk.uk.willdata.com
Address: 212.69.199.183
Domain Internet Groper: A tool for system administrators; it issues DNS queries and formats/interprets the answers.... Quite popular (*allegedly!*) with hackers...

Usage: `dig [@global-server] [domain] [q-type] [q-class] {q-opt} {global-d-opt} host [@local-server] {local-d-opt} [ host [@local-server] {local-d-opt} [...]]`

dig @lizzie www.google.co.uk any

```plaintext
; <<>> DiG 9.3.1 <<>> @lizzie www.google.co.uk any
; (1 server found); global options: printcmd; Got answer:
; ->> HEADER<<- opcode: QUERY, status: NOERROR, id: 16774
; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
; QUESTION SECTION:
;www.google.co.uk. IN ANY
; ANSWER SECTION:
www.google.co.uk. 86399 IN CNAME www.google.com.
; Query time: 63 msec
; SERVER: 192.168.1.45#53(192.168.1.45)
; WHEN: Mon Feb 5 14:11:43 2007
; MSG SIZE rcvd: 62
```

Tools in Detail .....
>....

dig @lizzie www.google.com any

;<<>> DiG 9.3.1 <<>> @lizzie www.google.com any
;(1 server found); global options: printcmd; Got answer:
;; ->>>HEADER<<- opcode: QUERY, status: NOERROR, id: 60773
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 3

;; QUESTION SECTION:
;www.google.com.

;; ANSWER SECTION:

;; ADDITIONAL SECTION:
www.l.google.com. 149 IN A 66.249.93.104
www.l.google.com. 149 IN A 66.249.93.99
www.l.google.com. 149 IN A 66.249.93.147

;; Query time: 56 msec
;; SERVER: 192.168.1.45#53(192.168.1.45)
;; WHEN: Mon Feb 5 14:15:13 2007
;; MSG SIZE  rcvd: 100
WHOIS

Domain name:
  google.co.uk

Registrant:
  Google Inc

Registrant type:
  Non-UK Corporation

Registrant's address:
  1600 Amphitheatre Parkway
  Mountain View
  CA
  94043
  United States

Registrant's agent:
  Markmonitor Inc. t/a Markmonitor
  URL: http://www.markmonitor.com

Relevant dates:
  Registered on: 14-Feb-1999
  Renewal date: 14-Feb-2009
  Last updated: 17-Jan-2007

Registration status:
  Renewal request being processed.

Name servers:
  ns1.google.com
  ns2.google.com
  ns3.google.com
  ns4.google.com
Estimates bandwidth, latency and packet loss on network links.

This is a re-working of the “pathchar” utility, written by Van Jacobson and, like traceroute, is based on repeated packet transmission and TTL variation (it can use ICMP or UDP).

It is available for most “*nix” systems: It works for IPv4 & IPv6.

Traceroute (UDP) knows when it has found its target by using a port number beyond the “normal range”... when ICMP “port unreachable” is returned it’s there!

Pchar sends many packets, one hop at a time, with varying the sizes, until the target is reached or the path fails. It calculates the latency from the ICMP message response times, and the throughput per hop from the variance in response speeds. Collectively, this also gives the overall round-trip delay for the whole path.

It is not fool-proof; it’s traffic may not be allowed; it is not a “Holy Grail”; but it does give a good indication!
Tools in Detail

```
**Pchar - ./pchar www.google.co.uk**

pchar to www.l.google.com (66.249.93.104) using UDP/IPv4
Using raw socket input
Packet size increments from 32 to 1500 by 32
46 test(s) per repetition : 32 repetition(s) per hop
Warning: target host did not respond to initial test.
0: 192.168.1.231 (dhcp-192-168-1-231.uk.willdata.com)
  Partial loss: 0 / 1472 (0%)
  Partial char: rtt = 0.959029 ms, (b = 0.001150 ms/B), r2 = 0.999475
  stddev rtt = 0.003212, stddev b = 0.000004
  Partial queueing: avg = 0.000171 ms (148 bytes)
  Hop char: rtt = 0.959029 ms, bw = 6954.330709 Kbps
  Hop queueing: avg = 0.000171 ms (148 bytes)
1: 81.144.212.33 (81.144.212.33)
  Partial loss: 0 / 1472 (0%)
  Partial char: rtt = 5.784087 ms, (b = 0.005317 ms/B), r2 = 0.999798
  stddev rtt = 0.009218, stddev b = 0.000011
  Partial queueing: avg = 0.002336 ms (667 bytes)
  Hop char: rtt = 4.825058 ms, bw = 1919.855256 Kbps
  Hop queueing: avg = 0.002165 ms (519 bytes)
2: 62.7.96.41 (62.7.96.41)
  Partial loss: 0 / 1472 (0%)
  Partial char: rtt = 5.824306 ms, (b = 0.005317 ms/B), r2 = 0.999847
  stddev rtt = 0.008008, stddev b = 0.000010
  Partial queueing: avg = 0.001486 ms (667 bytes)
  Hop char: rtt = 0.040220 ms, bw = --.--.-- Kbps
  Hop queueing: avg = -0.000850 ms (0 bytes)
3: 194.72.3.66 (core2-gig10-1.kingston.ukcore.bt.net)
  ??? - process hangs at this point!
```

This example shows a “pchar” test across a path where icmp responses are **not** allowed.
Tools in Detail . . . .

**Pchar - ./pchar 192.168.1.8 (a local address)**

pchar to 192.168.1.8 (192.168.1.8) using UDP/IPv4
Using raw socket input
Packet size increments from 32 to 1500 by 32
46 test(s) per repetition : 32 repetition(s) per hop

0: 192.168.1.231 (dhcp-192-168-1-231.uk.willdata.com)

- Partial loss: 0 / 1472 (0%)
- Partial char: rtt = 10.792415 ms, (b = 0.003369 ms/B), r2 = 0.157013
- stddev rtt = 0.950840, stddev b = 0.001177
- Partial queueing: avg = 0.015037 ms (4463 bytes)

1: 192.168.1.8 (zplex.uk.willdata.com)

- Path length: 1 hops
- Path char: rtt = 10.792415 ms r2 = 0.157013
- Path bottleneck: 2374.706954 Kbps
- Path pipe: 3203 bytes
- Path queueing: average = 0.015037 ms

Start time: Thu Feb 1 09:07:32 2007
End time: Thu Feb 1 09:14:22 2007

---

Partial loss = number of pkts / percentage pkts lost
Partial char = RTT, delay Byte, min delay pkt
Partial queueing = ave. queue of data incl. of this hop
Hop char = RTT and b/width for the current hop
Hop queueing = average queue of data this hop
Path bottleneck = “bottleneck” (achieved) bandwidth
Path pipe = Bandwidth-Delay Product = traffic “on the wire” (cf RWIN buffer)
Remember:

ICMP may be restricted over the test path

Not all platforms have the same controls or defaults

Think of the impact on the network of using these kind of tools!!

The figures produced are estimates (ref. pchar “man pages” of pchar and, as already mentioned for some previous tools, the results will probably not reflect the exact behaviour of the applications using the same path.

Learn more at:

http://www.kitchenlab.org/www/bmah/Software/pchar/
Netcat - a read/write utility for networks (TCP or UDP). It can be used on its own or be driven by user code. It is also a very powerful network debugging and exploration tool, which can create almost any kind of connection:-

- Outbound or inbound, TCP or UDP, to or from any ports
- Full DNS forward/reverse checking, with appropriate warnings
- Ability to use any local source port
- Ability to use any locally-configured network source address
- Built-in port-scanning capabilities, with randomizer
- Can read command line arguments from standard input
- Slow-send mode, one line every N seconds
- Hex dump of transmitted and received data
- Ability to let another program service established connections
- Telnet-options responder

Good for testing applications and application paths, but does not "test" or measure the network itself. **Beware of misuse!**
**Netcat**

connect to somewhere:  nc [-options] hostname port[s] [ports] ...
listen for inbound:     nc -l -p port [options] [hostname] [port]

options:
- **-d**     detach from console, background mode
- **-e prog**    inbound program to exec [dangerous!!]
- **-g gateway**  source-routing hop point[s], up to 8
- **-G num**     source-routing pointer: 4, 8, 12, ...
- **-h**         this help
- **-i secs**    delay interval for lines sent, ports scanned
- **-l**         listen mode, for inbound connects
- **-L**         listen harder, re-listen on socket close
- **-n**         numeric-only IP addresses, no DNS
- **-o file**    hex dump of traffic
- **-p port**    local port number
- **-r**         randomize local and remote ports
- **-s addr**    local source address
- **-t**         answer TELNET negotiation
- **-u**         UDP mode
- **-v**         verbose [use twice to be more verbose]
- **-w secs**    timeout for connects and final net reads
- **-z**         zero-I/O mode

port numbers can be individual or ranges: mm--nn [inclusive]

Learn more at:  
Tools in Detail . . . . .

Netcat - Retrieve page from web server

C:\>nc -v www.google.co.uk 80
www.l.google.com [216.239.59.103] 80 (http) open
GET / HTTP/1.0

HTTP/1.0 302 Found
Location: http://www.google.co.uk/
Cache-Control: private
Set-Cookie:
PREF=ID=bebf53d3e8c044c6:TM=1170500572:LM=1170500572:S=DBxO29wrWXh5ex5E;
expires=Sun, 17-Jan-2038 19:14:07 GMT
MT; path=/; domain=.google.com
Content-Type: text/html
Server: GWS/2.1
Content-Length: 221
Date: Sat, 03 Feb 2007 11:02:52 GMT
Connection: Keep-Alive

<html><head><meta http-equiv="content-type" content="text/html; charset=utf-8">
<title>302 Moved</title></head><body>
<h1>302 Moved</h1>
The document has moved
<a href="http://www.google.co.uk/">here</a>.
</body></html>
**Tools in Detail . . . .**

**Netcat - “NC” to “NC” connection**

```
c:\>nc -l -p 23 -t -e cmd.exe
```

```
C:\>nc 192.168.27.10 23
```

```
Microsoft Windows XP [Version 5.1.2600] . . .
```

```
C:\>ipconfig
```

```
Windows IP Configuration
Ethernet adapter Local Area Connection:
    Connection-specific DNS Suffix . . .
    IP Address . . . . . . . . . . . . . . . . . . : 192.168.27.10
    Subnet Mask . . . . . . . . . . . . . . . . . . : 255.255.255.0
    Default Gateway . . . . . . . . . . . . . . . . : 192.168.27.1
```

```
C:\>^C
```

```
C:\>ipconfig
```

```
Windows IP Configuration
Ethernet adapter Local Area Connection:
    Connection-specific DNS Suffix . . .
    IP Address . . . . . . . . . . . . . . . . . . : 192.168.27.50
    Subnet Mask . . . . . . . . . . . . . . . . . . : 255.255.255.0
    Default Gateway . . . . . . . . . . . . . . . . : 192.168.27.1
```

```
C:\>ipconfig
```

```
Windows IP Configuration
Ethernet adapter Local Area Connection:
    Connection-specific DNS Suffix . . .
    IP Address . . . . . . . . . . . . . . . . . . : 192.168.27.10
    Subnet Mask . . . . . . . . . . . . . . . . . . : 255.255.255.0
    Default Gateway . . . . . . . . . . . . . . . . : 192.168.27.1
```
SNMP - MIBs

Learn more at:
http://www.ireasoning.com/
Tools in Detail . . . . .

Packet Analysers – “Sniffers”

- “Original” capture routine - **TCPDUMP** + **LIBPCAP** (the Promiscuous Capture Library) or **WinPcap**. Available on most "open" platforms.

- **SSLDUMP** is TCPDUMP with SSL decryption capability.

- **ETHEREAL** is a packet analyzer based on TCPDUMP.

- **WIRESHARK** is the latest incarnation of ETHEREAL Shows actual packets on the network with “breakdown”. Good for true analysis of the network **and** for establishing "common use“ baselines.

- **EXIGENCE** provides similar functionality for z/OS.
Tools in Detail......

"Wireshark"

The three panes show the traffic flow, the headers, and the data in dump format.

Highlighting is reflected in the lower panes.

This image shows the IP header . . .
Tools in Detail . . . .

"Wireshark"

This image shows the UDP header . . .
This image shows the DATA; in this case a DNS Query.

( http://www.wireshark.org/ )
This image shows the equivalent displays in EXIGENCE; in this case for an FTP session. (http://www.willdata.com/)
Tools in Detail . . . .  

"ZEN Trace and Solve"

<table>
<thead>
<tr>
<th>TraceID</th>
<th>Status</th>
<th>Taken</th>
<th>Description</th>
<th>User</th>
<th>Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001</td>
<td>Taken</td>
<td>16 Feb 2011</td>
<td>Test Trace</td>
<td>TONYA</td>
<td>220</td>
</tr>
<tr>
<td>0002</td>
<td>Taken</td>
<td>16 Feb 2011</td>
<td>EE Trace</td>
<td>TONYA</td>
<td>178</td>
</tr>
<tr>
<td>0003</td>
<td>Taken</td>
<td>12 Feb 2011</td>
<td>Interface UNOS48</td>
<td>TONYA</td>
<td>18</td>
</tr>
<tr>
<td>0004</td>
<td>Taken</td>
<td>17 Feb 2011</td>
<td>Interface UNOS48</td>
<td>TONYA</td>
<td>24</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Packet Time</th>
<th>Packet Length</th>
<th>Protocol</th>
<th>IP Address</th>
<th>Window Seq No</th>
<th>ACK Seq No</th>
<th>Window Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>11:47:05.2131</td>
<td>64</td>
<td>TCP</td>
<td>10.5.1.11</td>
<td>6121 2499</td>
<td>BA874F89</td>
<td>32731</td>
</tr>
<tr>
<td>11:47:05.2482</td>
<td>86</td>
<td>TCP</td>
<td>10.5.1.11</td>
<td>6121 2499</td>
<td>39F0864F</td>
<td>32795</td>
</tr>
<tr>
<td>11:47:05.2842</td>
<td>78</td>
<td>TCP</td>
<td>10.20.1.133</td>
<td>50113 27000</td>
<td>8ABDAD90</td>
<td>32660</td>
</tr>
<tr>
<td>11:47:05.5701</td>
<td>217</td>
<td>TCP</td>
<td>192.168.5.103</td>
<td>255-255-255-255</td>
<td>137</td>
<td>137</td>
</tr>
<tr>
<td>11:47:05.5719</td>
<td>576</td>
<td>TCP</td>
<td>192.168.5.103</td>
<td>576</td>
<td>576</td>
<td></td>
</tr>
</tbody>
</table>

**TCP Header**
- Source Port: 27000
- Destination Port: 50113
- Sequence Number: 8ABDA90
- Ack Sequence Number: EFA5BA8A
- Header Length: 32 (8 words)
- Flags: PSH ACK
- Window Size: 32660
- TCP Checksum: 35662
- Urgent Pointer: 0

**Pocket Data**
Data Length: 165

```
+0000 465e5450 2c312e31 20320300 2045484d *HTTP/1 1 200 OK *
+0010 08461274 553a2057 65642c20 31352016 * Data: 16 F *
+0020 65622032 03313120 31313a34 373a3035 * eb 2011 11:47 05 *
+0030 20474d4f 0d645366 72766572 2a204168 * GET Server Ze *
+0040 48686973 4265616c 696e676573 5472616e67656e746573 * no-cache Comm *
+0050 3a20466f 0c666572 72656174 4a6f6e6564 65726573 * text/html Com *
+0060 74686573 74686520 4a6f6e6564 65726573 * no-cache Comm *
+0070 3a207468 656d7074 2d4c6f6e 67746f677973 * text/html Com *
+0080 74657374 2d546865 67746f6779 7320313811 * text-Content 181 *
+0090 2d32 *2 *
```
Tools in Detail . . . . .

“ZEN Trace and Solve”

ZTS - Exigence in the ZEN Framework.

( http://www.willdata.com/ )
And, In Passing . . . .

Network & Security testers

“Nessus” - (“The Tenable Newt”) a security vulnerability scanner.
(www.nessus.org)

“Insecure.org & nmap.org”

“Nmap” - a network and security scanner
(www.nessus.org & nmap.org)

Use responsibly – Use with care!
Nmap (edited)

```
> nmap -v -A 192.168.27.50
Starting Nmap 4.20 ( http://insecure.org ) at 2007-02-03 11:40 GMT Standard Time
Initiating ARP Ping Scan at 11:40
Scanning 192.168.27.50 [1 port]
Completed ARP Ping Scan at 11:40, 0.20s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 11:40
Completed Parallel DNS resolution of 1 host. at 11:40, 0.03s elapsed
Initiating SYN Stealth Scan at 11:40 : Scanning 192.168.27.50 [1697 ports]
Discovered open port 135/tcp on 192.168.27.50
Completed SYN Stealth Scan at 11:40, 39.05s elapsed (1697 total ports)
Initiating Service scan at 11:40 : Scanning 1 service on 192.168.27.50
Completed Service scan at 11:41, 11.63s elapsed (1 service on 1 host)
Warning: OS detection for 192.168.27.50 will be MUCH less reliable because we did not find at least 1 open and 1 closed TCP port
... Host 192.168.27.50 appears to be up ... good.
Interesting ports on 192.168.27.50:
Not shown: 1696 filtered ports
PORT    STATE SERVICE VERSION
135/tcp open  msrpc Microsoft Windows RPC
MAC Address: xx:xx:xx:xx:xx:xx (Dell ESG Pcba Test)
Running (JUST GUESSING): Microsoft Windows 2000|XP (98%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop : TCP Sequence Prediction: Difficulty=0 (Trivial joke)
... OS and Service detection performed. Nmap finished: 1 IP address (1 host up) scanned in 67.000 seconds
Raw packets sent: 3517 (162.066KB) | Rcvd: 86 (4770B)
```
Problem Diagnosis . . .

Outline Steps:

- Check the stack – “ping” local loopback
- “ping” the remote host/server name
- “ping” with IPaddress – the DNS may be down
- If “ping” fails “traceroute” - find where it stops
- Use “netstat” to check the interface
- Check routing (is it as expected?)
- If ping works, try “telnet” (standard port 23)
- If “telnet” works try telnet to the application port
- If that works try the application
- Use “netstat” to check the connection exists
- Check your syslogs (remember USS ! “syslogd” !)
- Do you still have a failure? ... trace it!
Summary

• Know Your Network!

• Keep Up-to-Date Documentations & Diagrams!
  (most tools can be used for practice at any time)

• Know the Tools

• Plan Your Approach to Any Problem

• Stop, Look, and LISTEN!!
Thank you !