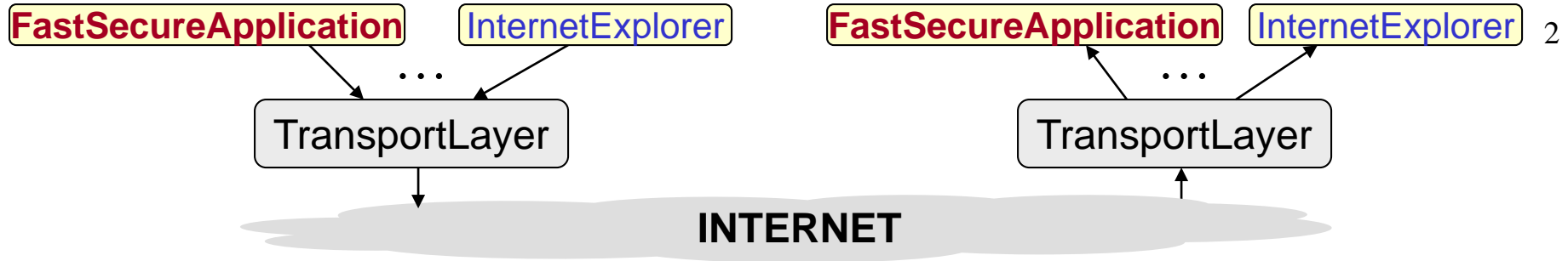


```
public void main(String[] args) {
    // GUI + file name input + file obj. fetching
    ...
    FastSecureApplication.send(file);
    ...
}
```

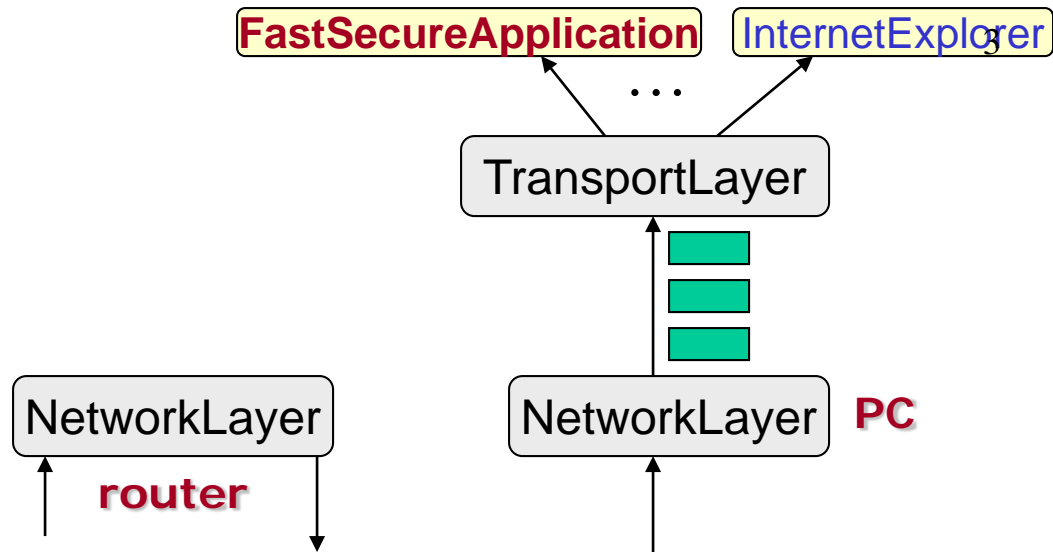
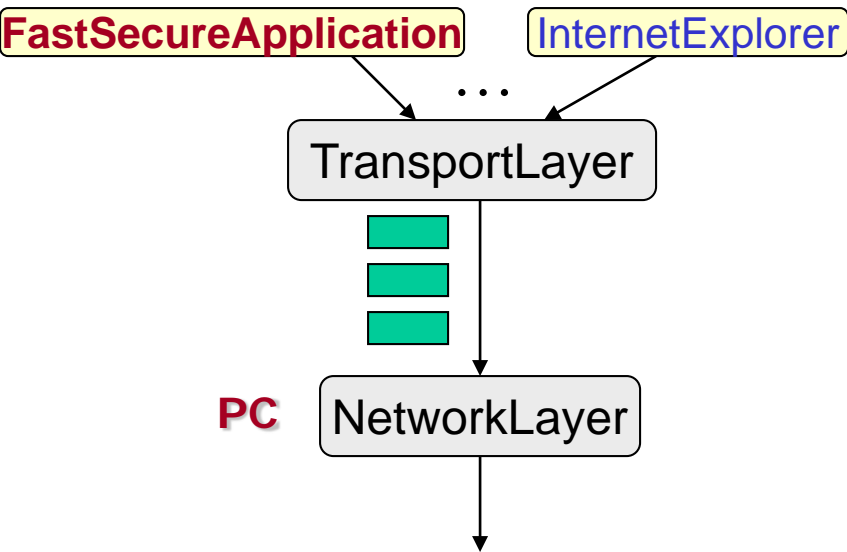
```
class FastSecureApplication {
    Object compress(Object file);
    Object encrypt(Object file);
    static void send(Object file);
    static Object deCompress();
    static Object deCrypt(Object file);
    static Object receive();
    ...
    static void send(Object file) {
        Object compressedFile = compress(file);
        Object encryptedFile = encrypt(compressedFile);
        TransportLayer.send(encryptedFile);
    }
    ...
}
```



```

class TransportLayer {
    int pickPortNumber();
    Packets[] reassemble(Object file);
    Packets[] addHeaders(Packets[] filePackets, int portNmb);
    static void send(Object applicationLayerFile);
    static Object assemble();
    static Packets[] removeHeaders(Object file);
    static void receive();
    ...
    static void send(Object applicationLayerFile) {
        int portNmb = pickPortNumber();
        Packets[] filePackets = reassemble(applicationLayerfile);
        Packets[] packetsWithHeader = addHeaders(filePackets, portNmb);
        for (int i=1; i < packetsWithHeader.length; i++) {
            NetworkLayer.send(packetsWithHeader[i]);
            ...
        }
    }
}

```



```

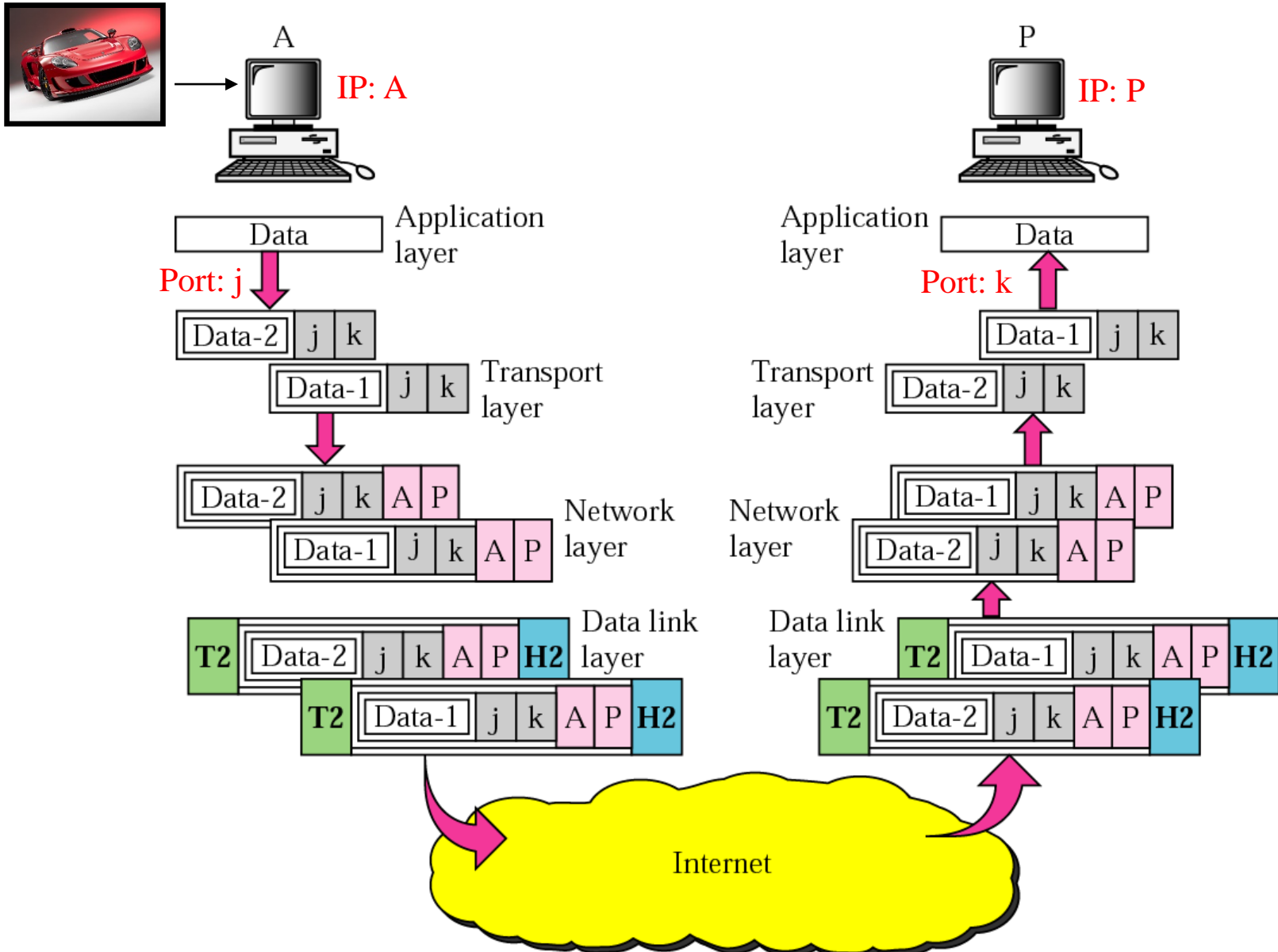
class NetworkLayer {
    ...
    static void send(Object transportLayerPacket) {
        Packet IPPacket = addHeader(transportLayerPacket, IPAddress, etc.);
        DataLinkLayer.send(IPPacket);
        ...
    }

    static void sendRouter(Object IPPacket) {
        ... findNextHop(IPPacket);
        DataLinkLayer.send(IPPacket);
        ...
    }
}

```

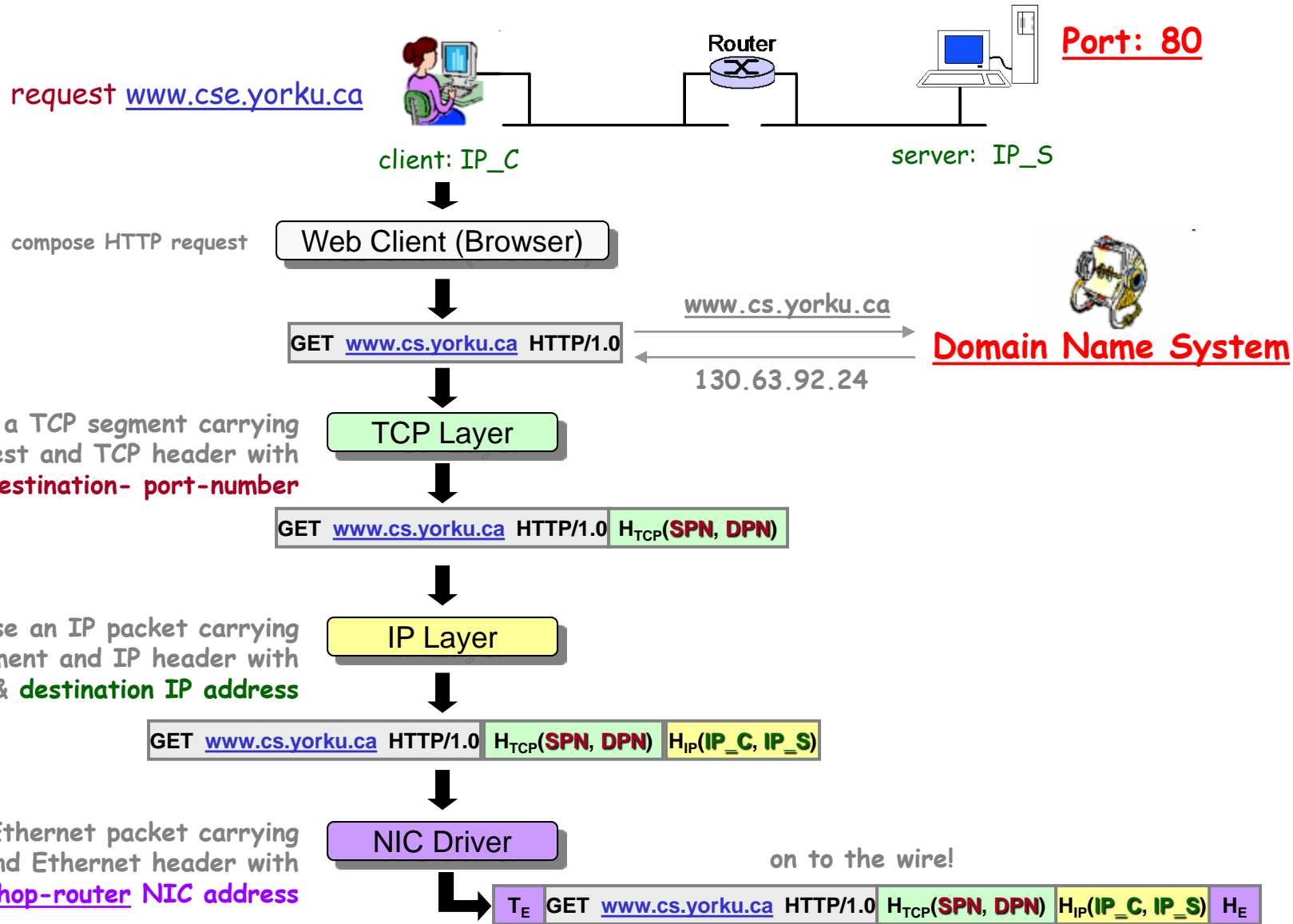
Example

Assume we want to exchange an image between computers A and P. The image, after being compressed, occupies 1000 bytes. The maximum TCP packet size is 500 bytes. How many packets are required in total, and how will these packets look like?

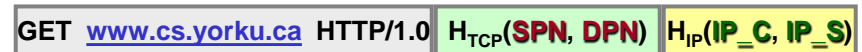
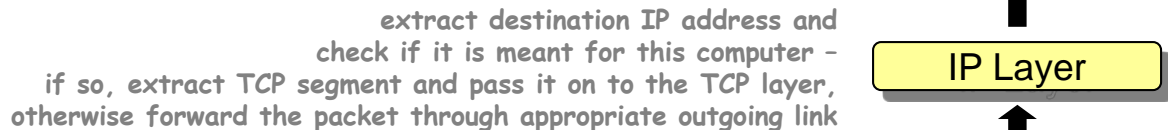
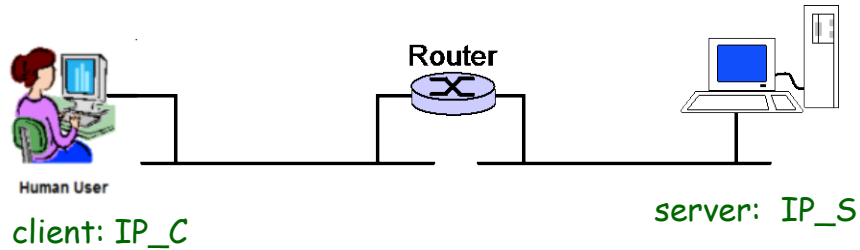


TCP/IP Protocol: How the Layers Work Together

Example [web-page retrieval – assumption: TCP connection established!]



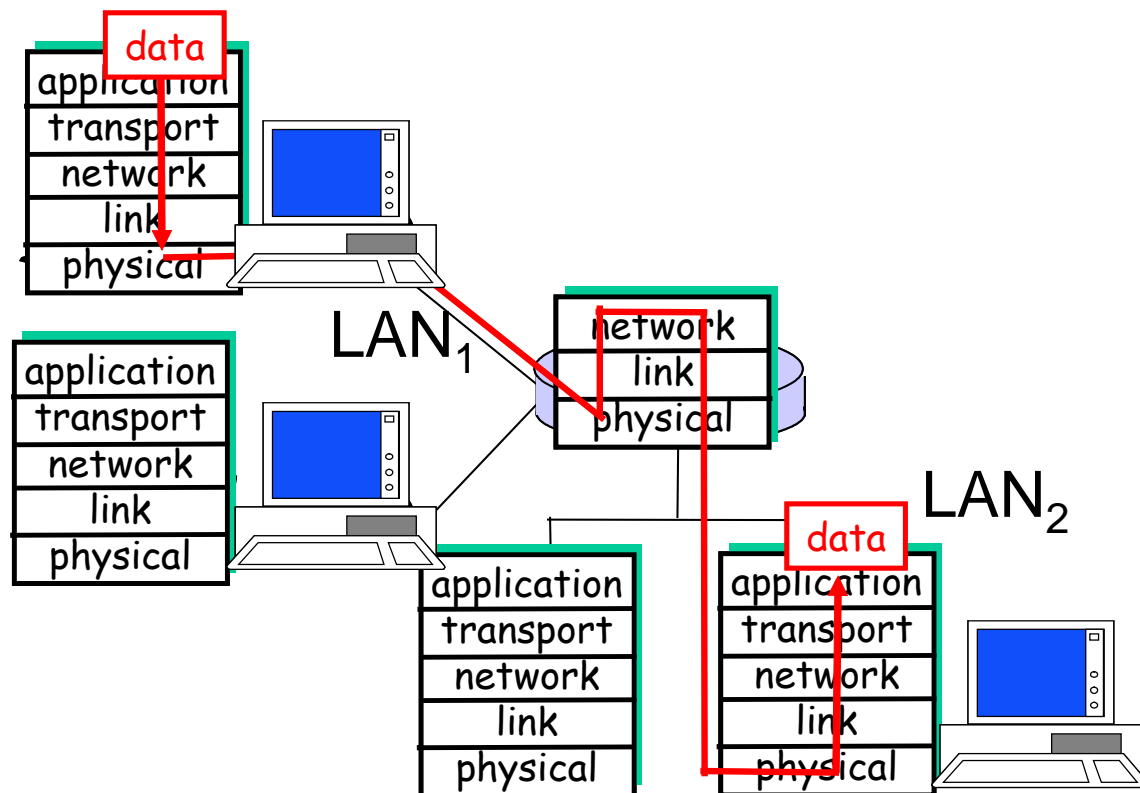
TCP/IP Protocol: How the Layers Work Together (cont.)



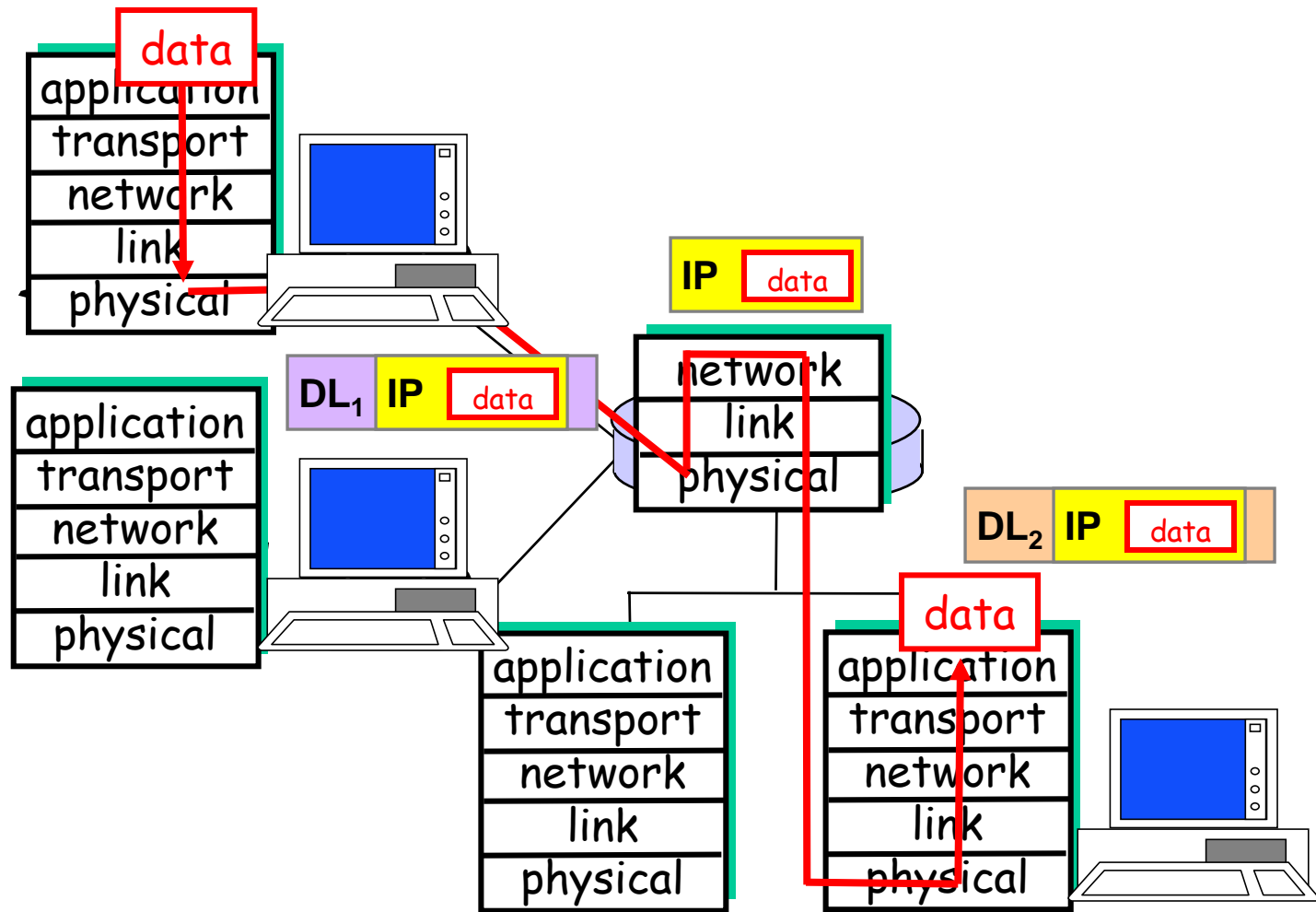
Bonus Question [layering – encapsulation]

Assume two computers, situated on two distant LANs - with different data-link technologies, communicate with each other over the Internet.

Does each of these computers have to be aware of the data-link technology / protocol run in the LAN of the other computer?



TCP/IP Protocol: How the Layers Work Together (cont.)

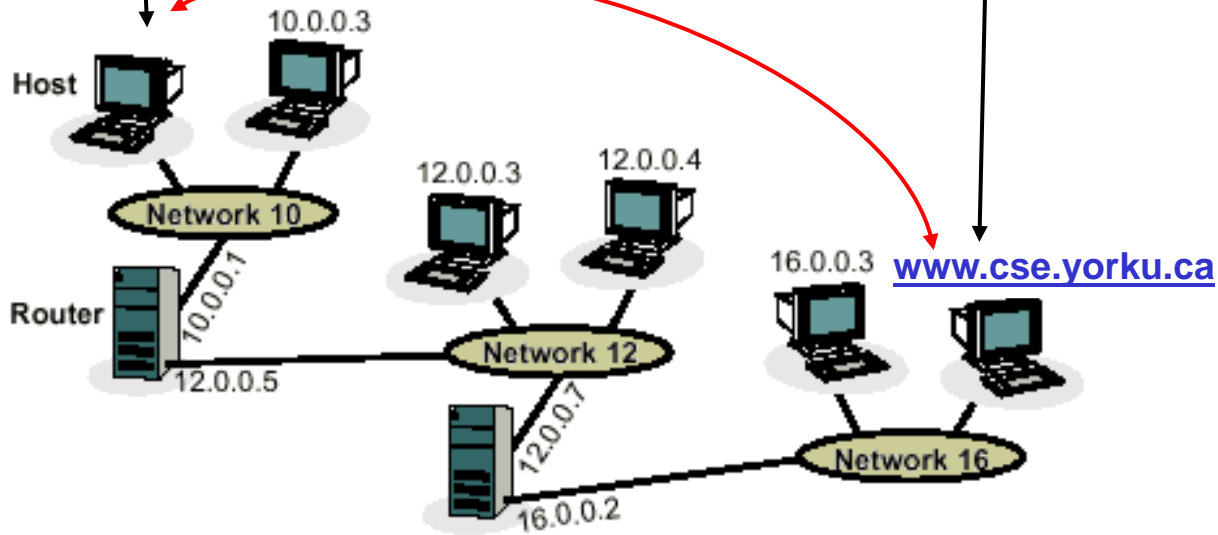


(Source: Kurose & Ross)

How to determine own IP & MAC address(es)?

How to determine the number and identity of intermediate routers?

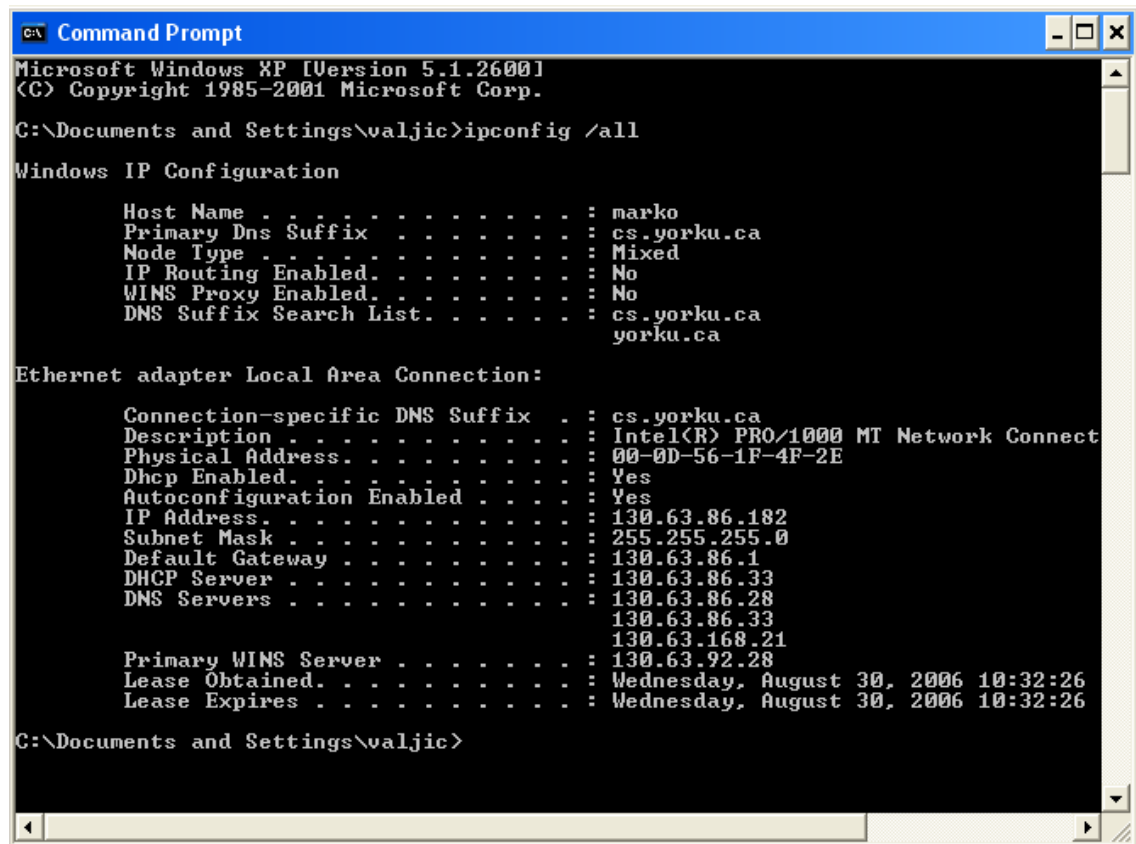
How to determine IP address of a remote machine, and whether it is up and running?



IP Utilities

IPCONFIG – Microsoft Windows OS tool; UNIX/Linux equivalents:
ifconfig, **ip addr**

- in simplest form returns IP address, subnet mask, default gateway
- **ipconfig /all** – returns above and DNS hostname, physical address, DNS and DHCP Server addresses, etc.



```
Command Prompt
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\valjic>ipconfig /all

Windows IP Configuration

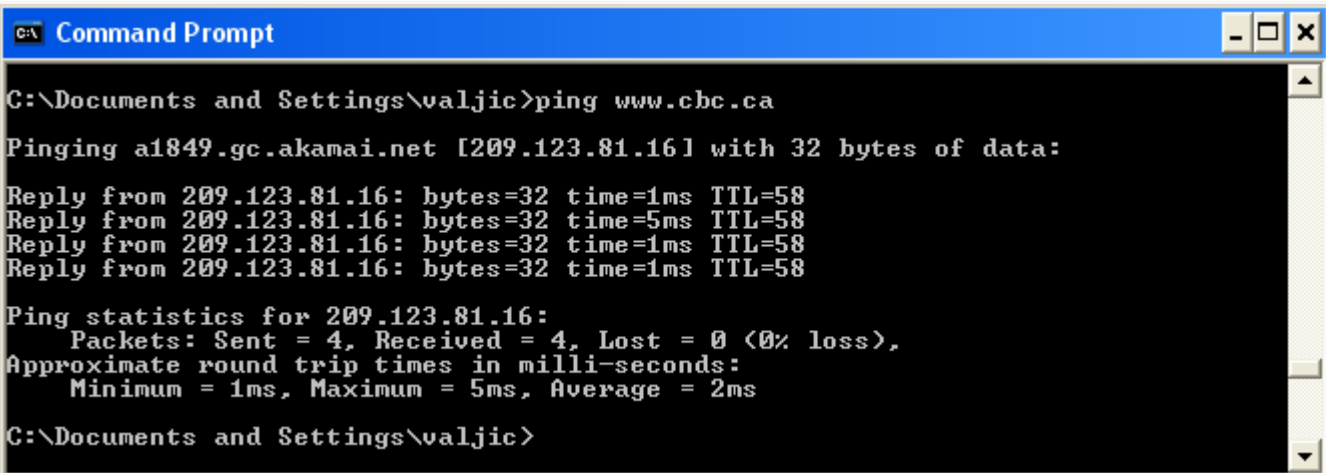
    Host Name . . . . . : marko
    Primary Dns Suffix . . . . . : cs.yorku.ca
    Node Type . . . . . : Mixed
    IP Routing Enabled. . . . . : No
    WINS Proxy Enabled. . . . . : No
    DNS Suffix Search List. . . . . : cs.yorku.ca
                                        yorku.ca

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix . : cs.yorku.ca
    Description . . . . . : Intel(R) PRO/1000 MT Network Connect
    Physical Address. . . . . : 00-0D-56-1F-4F-2E
    Dhcp Enabled. . . . . : Yes
    Autoconfiguration Enabled . . . . : Yes
    IP Address. . . . . : 130.63.86.182
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 130.63.86.1
    DHCP Server . . . . . : 130.63.86.33
    DNS Servers . . . . . : 130.63.86.28
                            130.63.86.33
                            130.63.168.21
    Primary WINS Server . . . . . : 130.63.92.28
    Lease Obtained. . . . . : Wednesday, August 30, 2006 10:32:26
    Lease Expires . . . . . : Wednesday, August 30, 2006 10:32:26

C:\Documents and Settings\valjic>
```

- PING** – standard troubleshooting tool (available on most OS) used to determine
- 1) whether a remote computer is currently “alive”
 - 2) round trip delay – max, min, average
- Windows *ping* sends 4 32-bit packets to destination and reports
 - a) how many packets reached another computer
 - b) roundtrip delay for each
 - *ping* makes use of **ICMP** messages
 - if host names used instead of IP addresses, ping relies on DNS service to obtain respective IP address



```
C:\Documents and Settings\valjic>ping www.chc.ca
Pinging a1849.gc.akamai.net [209.123.81.16] with 32 bytes of data:
Reply from 209.123.81.16: bytes=32 time=1ms TTL=58
Reply from 209.123.81.16: bytes=32 time=5ms TTL=58
Reply from 209.123.81.16: bytes=32 time=1ms TTL=58
Reply from 209.123.81.16: bytes=32 time=1ms TTL=58

Ping statistics for 209.123.81.16:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 5ms, Average = 2ms

C:\Documents and Settings\valjic>
```

Traceroute Origin – UNIX utility, but nearly all platforms have something similar

- Windows utility is called **tracert** – you can run tracert from MS-Dos Window, by entering tracert followed by domain name, e.g.

```
tracert www.cs.yourku.ca
```

- **tracert & traceroute have different implementation !**

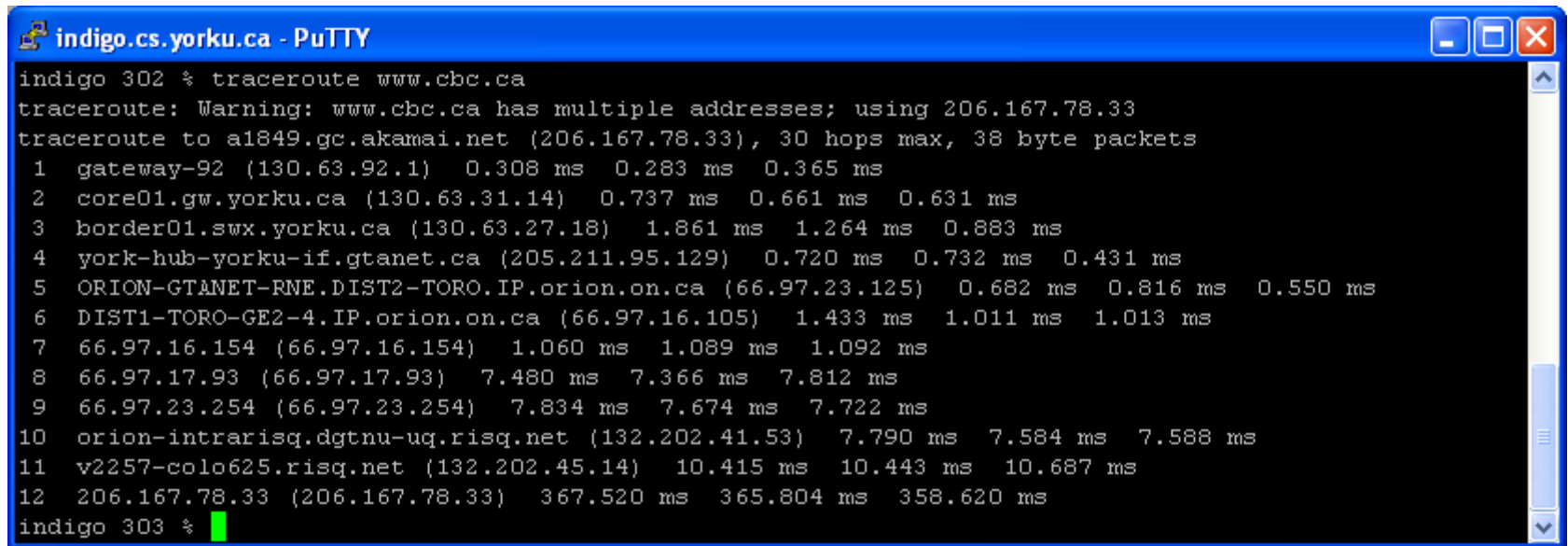
Traceroute Use – traceroute is generally used:

- (1) as network debugging tool by pinpointing network connectivity problems
- (2) for identifying IP addresses

Example [traceroute]

If you are visiting a Web site and pages are appearing slowly, you can use traceroute to figure out where the longest delay(s) are occurring.

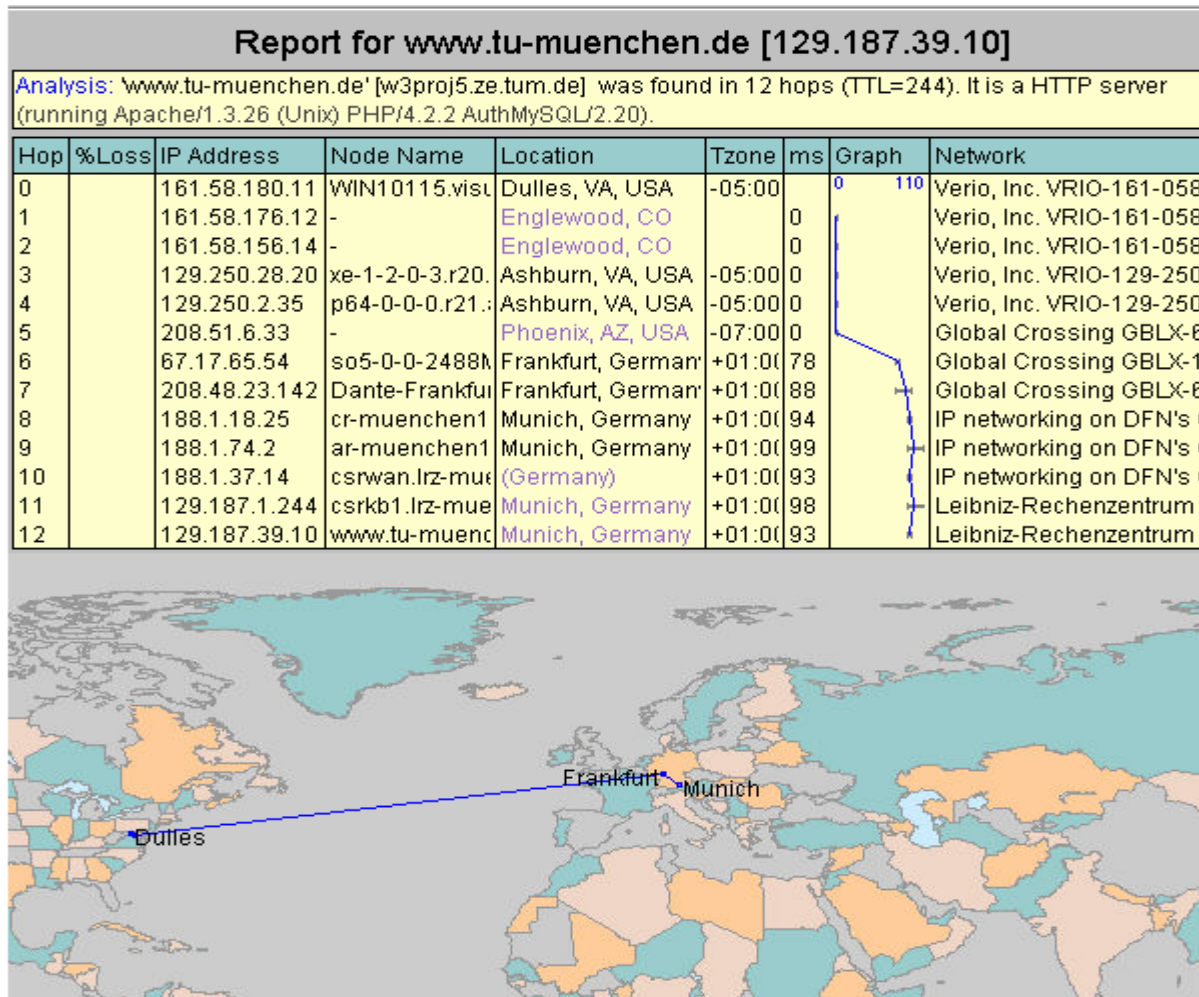
Example [traceroute www.bbc.co.uk]



The screenshot shows a PuTTY terminal window titled "indigo.cs.yorku.ca - PuTTY". The terminal displays the execution of the command "traceroute www.cbc.ca". The output shows a warning about multiple addresses for www.cbc.ca and then a detailed traceroute path to 206.167.78.33, listing 12 hops with their respective IP addresses and round-trip times.

```
indigo 302 % traceroute www.cbc.ca
traceroute: Warning: www.cbc.ca has multiple addresses; using 206.167.78.33
traceroute to a1849.gc.akamai.net (206.167.78.33), 30 hops max, 38 byte packets
 1 gateway-92 (130.63.92.1)  0.308 ms  0.283 ms  0.365 ms
 2 core01.gw.yorku.ca (130.63.31.14)  0.737 ms  0.661 ms  0.631 ms
 3 border01.swx.yorku.ca (130.63.27.18)  1.861 ms  1.264 ms  0.883 ms
 4 york-hub-yorku-if.gtanet.ca (205.211.95.129)  0.720 ms  0.732 ms  0.431 ms
 5 ORION-GTANET-RNE.DIST2-TORO.IP.orion.on.ca (66.97.23.125)  0.682 ms  0.816 ms  0.550 ms
 6 DIST1-TORO-GE2-4.IP.orion.on.ca (66.97.16.105)  1.433 ms  1.011 ms  1.013 ms
 7 66.97.16.154 (66.97.16.154)  1.060 ms  1.089 ms  1.092 ms
 8 66.97.17.93 (66.97.17.93)  7.480 ms  7.366 ms  7.812 ms
 9 66.97.23.254 (66.97.23.254)  7.834 ms  7.674 ms  7.722 ms
10 orion-intrarisq.dgtnu-uq.risq.net (132.202.41.53)  7.790 ms  7.584 ms  7.588 ms
11 v2257-colo625.risq.net (132.202.45.14)  10.415 ms  10.443 ms  10.687 ms
12 206.167.78.33 (206.167.78.33)  367.520 ms  365.804 ms  358.620 ms
indigo 303 %
```

VisualRoute for Internet Performance: <http://visualroute.visualware.com/>



<http://www.visualware.com/resources/tutorials/tracert.html>

Top-down network design - Google Books - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites Media

Address <http://books.google.ca/books?id=YKWfuGG5mXMC&pg=PA302&pg=PA302&dq=traceroute+shorter+RTT+time&source=...> Go Links

Contents Page 302 Link Feedback

Result 1 of 1 in this book for **traceroute shorter RTT time** Clear search

NOTE

Traceroute is used to determine the IP routing path to a remote device. With UNIX and Cisco IOS operating systems, a **traceroute** packet is a User Datagram Protocol (UDP) "probe" packet sent to a high port number, in the 33,000 to 43,000 range. Microsoft operating systems send a ping rather than a UDP packet. **Traceroute** works by taking advantage of the ICMP error message a router generates when a packet exceeds its **time-to-live (TTL)** value. TTL is a field in the IP header of an IP packet.

Traceroute starts by sending a UDP probe or ping packet with a TTL of one. This causes the first router in the path to discard the packet and send back a **time-exceeded (TTL exceeded)** ICMP message. The **traceroute** command then sends several packets, increasing the TTL by one after a few packets have been sent at each TTL value. For example, it sends a few packets with TTL equal to 1, then a few packets with TTL equal to 2, then a few packets with TTL equal to 3, and so on, until the destination host is reached.

Each router in the path decrements the TTL. The router that decrements the TTL to zero sends back the **time-exceeded (TTL exceeded)** message. The final destination host sends back a ping reply (if the sender was using a Microsoft operating system) or a destination unreachable (port-unreachable) ICMP message (if the sender was using UNIX or Cisco IOS), because the high UDP port number is not a well-known port. This process allows a user to see a message from every router in the path to the destination, and a message from the destination.

Unfortunately, **traceroute** is not dependable. Some routers do not send back **time-exceeded** messages, either because they are simply not programmed to do so, or because they are configured to rate-limit ICMP or block ICMP for security reasons. Some routers incorrectly use the TTL of the incoming packet to send the **time-exceeded** message, which does not work. Also, some systems do not send the port-unreachable message, which means that **traceroute** waits for a long **time** before timing out. Finally, some service providers purposely change the results of **traceroute** to hide internal hops so that users think the providers' paths must be **shorter** than competitors' paths.

Fault Management

Fault management refers to detecting, isolating, diagnosing, and correcting problems. It also includes processes for reporting problems to end users and managers, and tracking trends related to problems. In some cases, fault management means developing workarounds until a problem can be fixed.

Network users expect quick and reliable fault resolution. They also expect to be kept informed about ongoing problems and to be given a timeframe for resolution. After a problem is resolved, they expect the problem to be tested and then documented in some sort

Copyrighted material

Done Internet

- Q.1** Which layer provides logical addressing that routers will use for path determination?
- A.1** Network Layer
- Q.2** Which layer is responsible for converting data packets into electrical signal?
- A.2** Physical Layer
- Q.3** Which layer combines bits into bytes and bytes into frames, uses MAC addressing, and provides error detection?
- A.3** Data-link Layer
- Q.4** Which layer is used for reliable communication between end nodes over a WAN and controlling the flow of information?
- A.4** Transport Layer

Q.5 Which fields are contained within an IEEE Ethernet frame header?

(a) Source and destination MAC address.

(b) Source and destination network (IP) address.

(c) Source and destination MAC address and source and destination network (IP) address.

Q.6 When data is encapsulated, which is the correct order?

(a) Data, frame, packet, segment, bit.

(b) Segment, data, packet, frame, bit.

(c) Data, segment, packet, frame, bit.

(d) Data, segment, frame, packet, bit.