

Outline

- ❖ Introduction
- ❖ IP Address & MAC Address
- ❖ TCP/UDP/ICMP
- ❖ **IP Gateway, Network Mask, TTL**
- ❖ Routing Protocol
- ❖ Network Address Translation (NAT)
- ❖ Domain Name System (DNS)
- ❖ Dynamic Host Configuration Protocol (DHCP)
/ Asymmetric Digital Subscriber Line (ADSL)
- ❖ HyperText Transfer Protocol (HTTP)
Protocol
- ❖ Virtual Private Network (VPN)

Gateway

❖ LAN 對 WAN 之出入口

- 一般而言，Gateway就如同區域網路的出口，若有資料要傳送到Internet上，均需透過它，再經過中間的Router轉送到其他Router上，最後再到達目的端。

❖ 主要的功能是連接不同的網路架構，而且必須視架構不同而轉換不同的通訊網路協定層級

- ❖ 用來連接不同的通訊協定網路，例如：
TCP/IP，**X.25**，**Systems Network Architecture (SNA)**等，Gateway必須將所接收的 **Packet**，轉換成目的位址之網路所能瞭解的資料格式(訊息格式轉換，位址轉換，以及協定轉換等)

❖ X.25

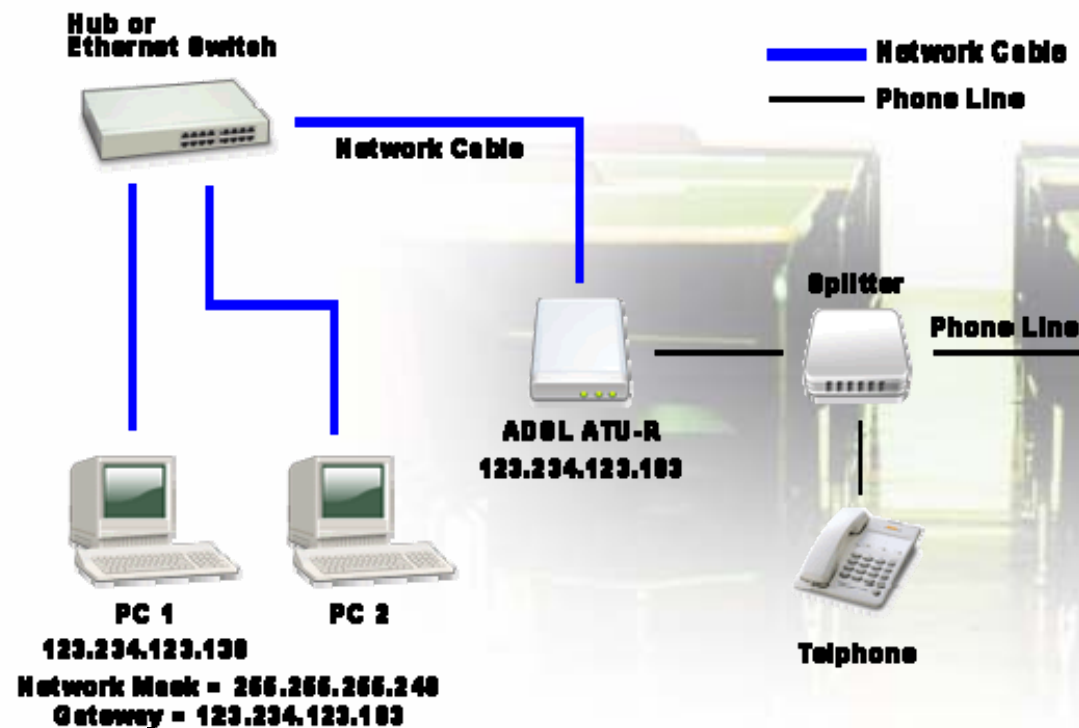
- 一種專用於資料終端設備與封包交換網路間的CCITT協定
- CCITT: Consultative Committee for International Telegraph and Telephone 國際電報電話諮詢委員會

❖ SNA

- IBM 主機系統所使用的通信協定

舉例說明

- ❖ 如下圖，**ADSL ATU-R**做為其他電腦上網的**Gateway**
- ❖ 假設**PC1**要傳送**Packet**到**Gateway**之流程：
 - 藉由PC1上Gateway的設定：藉ARP 找到Gateway 之 MAC Address
 - 由PC1 之**Netmask**，決定哪些目的IP 需要送到Gateway 轉出去



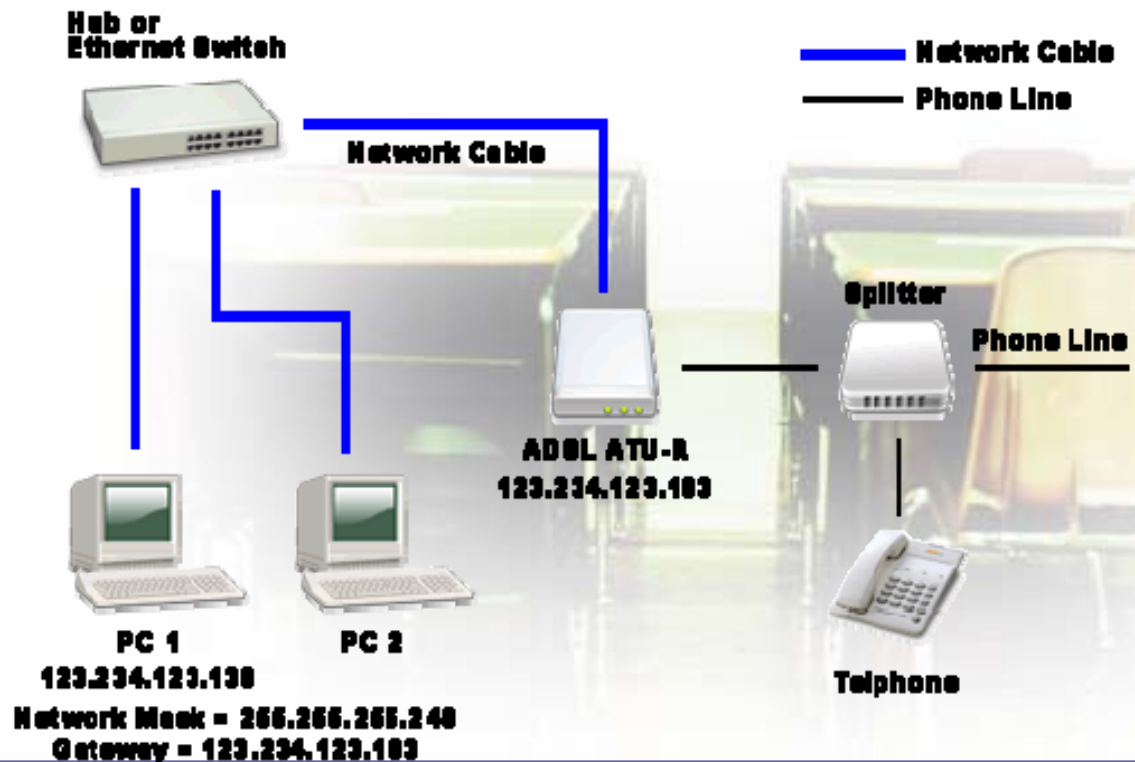
簡單驗證實驗

CYUT NC

ADSL ATU-R 當Gateway

❖ PC 1 是使用Windows2000作業系統執行以下的指令步驟：

1. arp -a
2. ping 168.95.1.1
3. arp -a
4. route print



執行結果說明

❖ arp -a

```
Microsoft Windows 2000 [版本 5.00.2195]  
(C) Copyright 1985-2000 Microsoft Corp.
```

```
C:\>arp -a
```

```
No ARP Entries Found
```

← The ARP table does not have any information prior to an Internet connection

```
C:\>
```

執行結果說明

❖ ping 168.95.1.1

```
Microsoft Windows 2000 [版本 5.00.2195]  
(C) Copyright 1985-2000 Microsoft Corp.
```

```
C:\>arp -a  
No ARP Entries Found
```

```
C:\>
```

```
Pinging 168.95.1.1 with 32 bytes of data:
```

```
Reply from 168.95.1.1: bytes=32 time=40ms TTL=249  
Reply from 168.95.1.1: bytes=32 time=30ms TTL=249  
Reply from 168.95.1.1: bytes=32 time=30ms TTL=249  
Reply from 168.95.1.1: bytes=32 time=30ms TTL=249
```

```
Ping statistics for 168.95.1.1:
```

```
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),  
Approximate round trip times in milli-seconds:  
    Minimum = 30ms, Maximum = 40ms, Average = 32ms
```

```
C:\>_
```

TTL value is related to the routers which have been passed through



執行結果說明

❖ arp -a

```
C:\>arp -a  
No ARP Entries Found
```

```
C:\>
```

```
Pinging 168.95.1.1 with 32 bytes of data:
```

```
Reply from 168.95.1.1: bytes=32 time=40ms TTL=249  
Reply from 168.95.1.1: bytes=32 time=30ms TTL=249  
Reply from 168.95.1.1: bytes=32 time=30ms TTL=249  
Reply from 168.95.1.1: bytes=32 time=30ms TTL=249
```

```
Ping statistics for 168.95.1.1:
```

```
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),  
    Approximate round trip times in milli-seconds:  
        Minimum = 30ms, Maximum = 40ms, Average = 32ms
```

```
C:\>arp -a
```

```
Interface: 123.234.123.203 on Interface 0x1000004  
    Internet Address      Physical Address      Type  
    123.234.123.193      00-d0-59-a5-d1-76    dynamic
```

The MAC Address of the gateway network interface

執行結果說明

❖ route print

```
123.234.123.193      00-d0-59-a5-d1-76      dynamic

C:\>route print
=====
Interface List
0x1 ..... MS TCP Loopback interface
0x2 ...44 45 53 54 77 77 ..... NIS PPPoE Adapter #1
0x1000004 ...00 e0 98 81 16 0d ..... Fast Ethernet PC Card
=====
Active Routes:
Network Destination        Netmask          Gateway          Interface        Metric
0.0.0.0                    0.0.0.0          123.234.123.193  123.234.123.203    1
123.234.123.192            255.255.255.240  123.234.123.203  123.234.123.203    1
123.234.123.203            255.255.255.255  127.0.0.1        127.0.0.1          1
123.255.255.255            255.255.255.255  123.234.123.203  123.234.123.203    1
127.0.0.0                  255.0.0.0        127.0.0.1        127.0.0.1          1
224.0.0.0                  224.0.0.0        123.234.123.203  123.234.123.203    1
255.255.255.255            255.255.255.255  123.234.123.203  2                    1
Default Gateway:          123.234.123.193
=====
Persistent Routes:
None

C:\>
```

All the packets' destination IP which is not listed in routing table will be sent to 123.234.123.193



Routing Table的意義

CYUT NC

Routing Table

- ❖ 當 **Router** 設備接收到 **Packet** 之後，它會依照 **Packet** 的目的地 **IP (Destination IP Address)** 決定要對 **Packet** 做怎樣的動作
 - 諸如從某個網路介面轉送、忽略或是傳回不予轉送的訊息；而做這些決策的依據，就是 **Routing Table**。
- ❖ **Routing Table** 可能以不同資料結構存在於所有具有 **Layer 3** 網路以上功能的設備中，包含 **PC**、**Firewall**、**Router**，甚至有網路功能的 **PDA**。

Linux Host Routing Table範例 (ip route)

- ❖ 例如以下是一 Linux Host 的 Routing Table，使用 `/sbin/ip route` (使用mask bit方式列) 命令，或使用 `/sbin/route`命令如下頁。

```
[root@gr ~]# /sbin/ip route
```

```
61.218.155.224/28      dev eth1      scope link  src 61.218.155.226
192.168.2.0/24        dev eth0      scope link  src 192.168.2.254
192.168.0.0/24      dev eth0      scope link  src 192.168.0.10
192.168.254.0/24     dev eth0      scope link  src 192.168.254.254
10.0.0.0/16           dev eth0      scope link  src 10.0.0.10
172.16.0.0/16         dev eth0      scope link  src 172.16.0.10
127.0.0.0/8           dev lo        scope link
default via 61.218.155.225 dev eth1
```

Linux Host Routing Table範例(route)

```
[root@gr ~]# /sbin/route
```

```
Kernel IP routing table
```

Destination	Gateway	Genmask	Metric	Iface
NAT				
61.218.155.224	*	255.255.255.240	0	eth1
192.168.2.0	*	255.255.255.0	0	eth0
192.168.0.0	*	255.255.255.0	0	eth0
192.168.254.0	*	255.255.255.0	0	eth0
10.0.0.0	*	255.255.0.0	0	eth1
172.16.0.0	*	255.255.0.0	0	eth0
127.0.0.0	*	255.0.0.0	0	lo
default	61.218.155.225	0.0.0.0	0	eth1

Routing Table Entry

- ❖ 一般Routing Table Entry 搜尋規則是根據IP Packet之目的IP (Destination IP) ，於Routing Table中以**Longest Prefix Length Match** 為原則進行查詢
 - 即當比對時從最前面比對過來符合長度最長的選擇法
- ❖ Default為**0.0.0.0 / 0** ，即任意 IP 皆符合條件。若合乎該Entry 的條件，則依照其指定的規則轉送 Packet

Routing Table Entry

- ❖ 若一Packet 進入其網路核心之 Destination IP 為 61.218.155.236 ，與第一個 Entry 的 61.218.155.224/28 從頭開始比對有28 bits相同，因此符合條件。

目的地IP 61.218.155.236
與以下 Routing Table entry
比對，決定從何介面送出

目的地 IP 二進位
00111101 11011010 10011011 1110 1100

Longest Prefix Length Match

Routing Table Entry

61.218.155.236

00111101 11011010 10011011 1110 1100

	Routing Table Entry	Binary Format	Interface
✓ 1 Longest Prefix Length Match	61.218.155.224/28	<p><u>00111101 11011010 10011011 1110 0000</u></p> <p> ----- 28 bits ----- </p> <p><u>00111101 11011010 10011011 1110 1100</u></p> <p>28 prefix bits matched</p>	eth1
✓ 2	Suppose there is another entry with 61.218.0.0/16	<p><u>00111101 11011010 00000000 0000 0000</u></p> <p> ----- 16 bits ----- </p> <p><u>00111101 11011010 10011011 1110 1100</u></p> <p>16 prefix bits matched</p>	eth2
3	Suppose there is another entry with 61.218.155.224/30	<p><u>00111101 11011010 10011011 1110 0000</u></p> <p> ----- 30 bits ----- </p> <p><u>00111101 11011010 10011011 1110 1100</u></p> <p>Not matched</p>	eth3
4	192.168.2.0/24	<p><u>11000000 10101000 00000010 00000000</u></p> <p><u>00111101 11011010 10011011 1110 1100</u></p> <p>Not matched</p>	eth0
✓ 5	Default 0.0.0.0/0 matches 0 bits	<p><u>00000000 00000000 00000000 00000000</u></p> <p><u>00111101 11011010 10101101 1110 1100</u></p> <p>Considered 0 bit matched</p>	eth1

Routing Table Entry

- ❖ 此 **Routing Table Entry** 告訴當網路核心進行 **Routing Decision** 時，目的地 **IP** 的前 **28 bits** 若是與 **61.218.155.224** 前面的 **28 bits** 相同，則經由介面 **eth1** 送出
- ❖ **scope link** 是說明符合條件的 **IP**，與其直接相連，其 **Destination MAC** 可藉由 **ARP** 得知，**src 61.218.155.226** 則是說明 **eth1** 介面使用 **IP** 為 **61.218.155.226**
- ❖ 結論：合乎條件共有 **1,2,5** 等三個，但是 **Entry 1** 是 **match** 最長的第一筆記錄 **28 bits**，因此由 **Entry 1** 的網路介面即 **eth1** 送出

Static & Dynamic Routing

❖ Static Routing

- 由系統管理員因網路所需，設置固定的路徑表，稱為靜態（static）路徑表
- 一般在系統安裝時就根據網路的配置情況預先設定

❖ Dynamic Routing

- Router根據網絡系統的運行情況而自動調整的路徑表
- 路由器根據Routing Protocol提供的功能，自動學習和記憶網路運行情況，在需要時自動計算數據傳輸的最佳路徑。

❖ TCP/IP中重要的routing protocol

- Routing Information Protocol (RIP), Open Shortest Path First (OSPF), Border Gateway Protocol (BGP) (Chapter 5)

ifconfig 與 Static Route Entry

- ❖ **ifconfig**指令除了設定**ip**及**netmask**之外，還會自動新增一筆對應的**static route entry**

```
bash# ip route
192.168.2.0/24 dev eth2 proto kernel scope link src 192.168.2.1
192.168.1.0/24 dev eth1 proto kernel scope link src 192.168.1.1
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.1
192.168.251.0/24 dev eth3 proto kernel scope link src 192.168.251.251
bash#
bash#
bash# ifconfig eth0:1 192.168.4.1 netmask 255.255.255.0
bash#
bash#
bash# ip route
192.168.4.0/24 dev eth0 proto kernel scope link src 192.168.4.1
192.168.2.0/24 dev eth2 proto kernel scope link src 192.168.2.1
192.168.1.0/24 dev eth1 proto kernel scope link src 192.168.1.1
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.1
192.168.251.0/24 dev eth3 proto kernel scope link src 192.168.251.251
bash# █
```

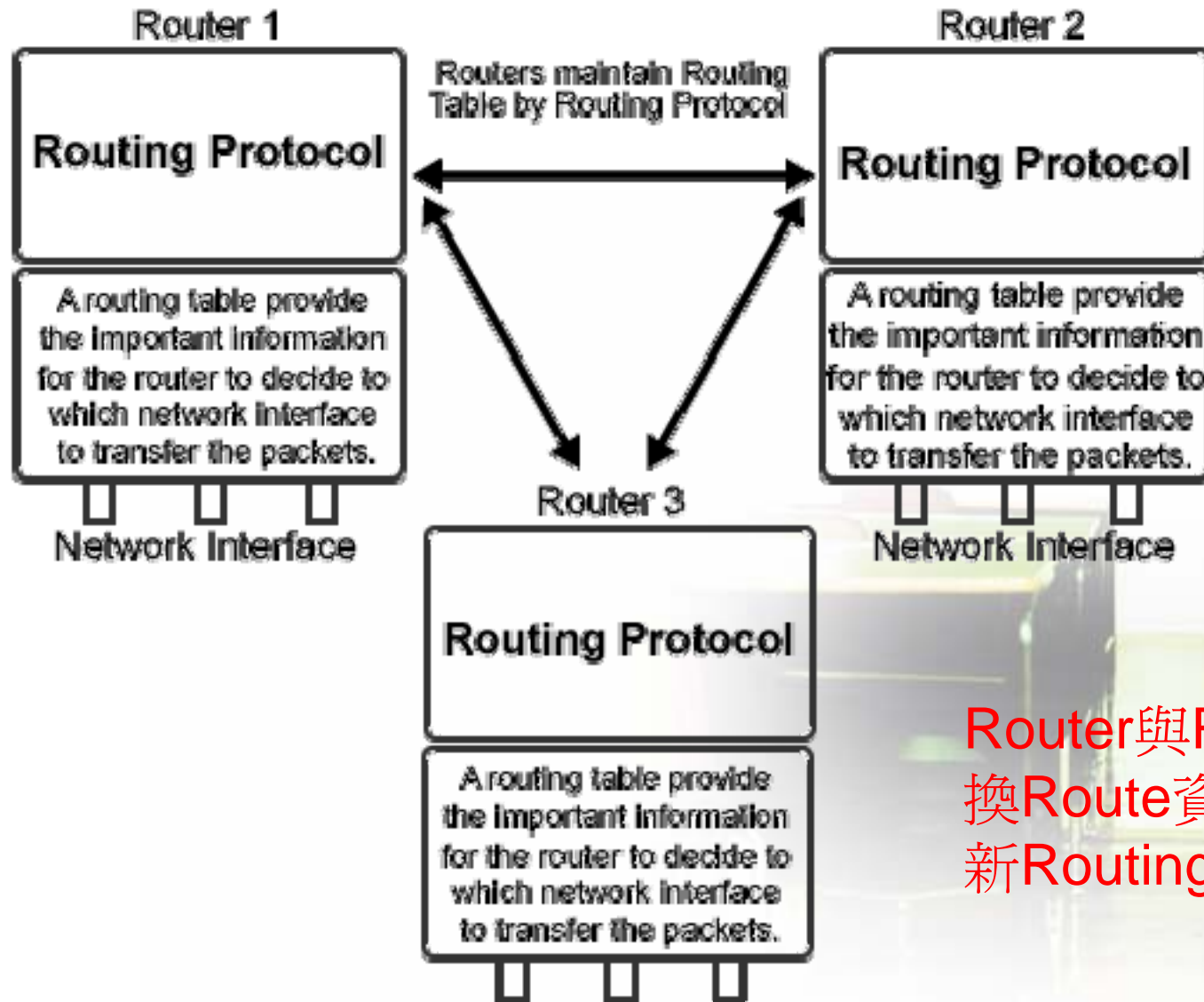
Routing的選擇

```
bash# ip route
192.168.2.0/24 dev eth2 proto kernel scope link src 192.168.2.1
192.168.1.0/24 dev eth1 proto kernel scope link src 192.168.1.1
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.1
192.168.251.0/24 dev eth3 proto kernel scope link src 192.168.251.251
bash#
bash# ping -c3 192.168.2.2
PING 192.168.2.2 (192.168.2.2) from 192.168.2.1 : 56(84) bytes of data.
64 bytes from 192.168.2.2: icmp_seq=1 ttl=64 time=0.319 ms
64 bytes from 192.168.2.2: icmp_seq=2 ttl=64 time=0.156 ms
64 bytes from 192.168.2.2: icmp_seq=3 ttl=64 time=0.152 ms

--- 192.168.2.2 ping statistics ---
3 packets transmitted, 3 received, 0% loss, time 2010ms
rtt min/avg/max/mdev = 0.152/0.209/0.319/0.077 ms
bash#
bash# arp -an
? (192.168.2.2) at 00:30:C7:81:44:21 [ether] on eth2
? (192.168.251.254) at 00:10:F3:03:16:B3 [ether] on eth3
? (192.168.251.22) at 00:E0:98:81:16:0D [ether] on eth3
bash#
bash# ip route del 192.168.2.0/24
bash#
bash# ip route
192.168.1.0/24 dev eth1 proto kernel scope link src 192.168.1.1
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.1
192.168.251.0/24 dev eth3 proto kernel scope link src 192.168.251.251
bash#
bash# ping 192.168.2.2
connect: Network is unreachable
bash#
```



Routing Table 與 Routing Protocol



Router與Router之間交換Route資訊，進而更新Routing Table

❖ Router的設計可分爲

- **Fast Path:** 需要快速查詢Routing Table並決策Packet如何轉送，故需使用硬體晶片或Network Processor設計加速Routing Table查詢。
- **Slow Path:** 速度不需太快，可用一般CPU處理Routing Protocol。

Gateway 和Router

❖ Gateway

- 較廣義的名詞，用來連接不同的網路，甚至不同的通訊協定，從第三層到第七層OSI皆有可能
- Gateway可以是Router、防火牆、Proxy，所以說Gateway 不一定是指Router

❖ Router

- 是指尋找、決定最佳的傳輸路徑，並根據這些路徑建立Routing Table轉送 Packet或者是過濾 Packet的網路裝置

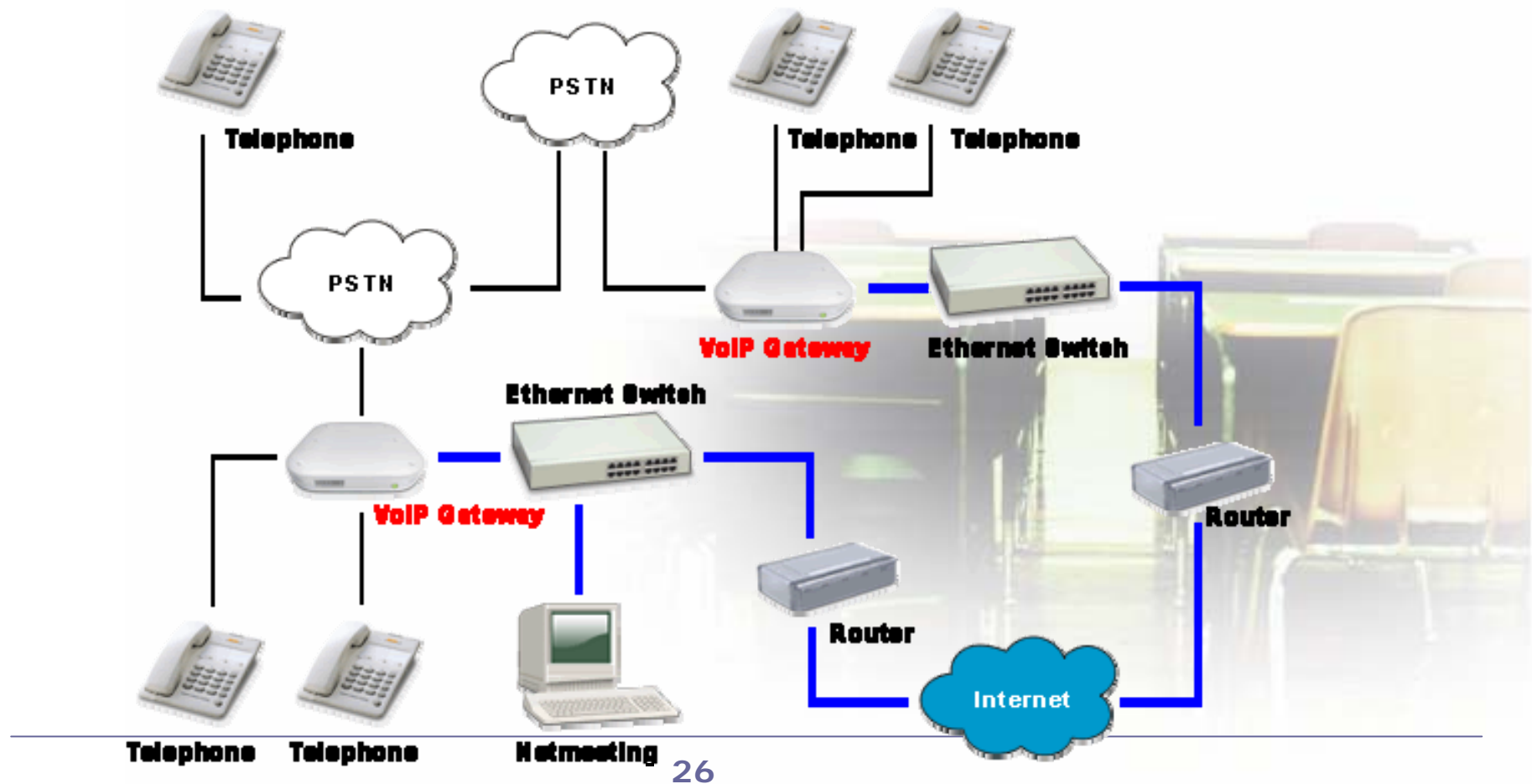
A photograph of a classroom. In the foreground, there are several rows of wooden desks with attached chairs. The desks are arranged in a grid pattern. In the background, a large green chalkboard is mounted on the wall. The text "Application Gateway" is written in white on the chalkboard. The text "CYUT NC" is written in white in the bottom right corner of the image.

Application Gateway

CYUT NC

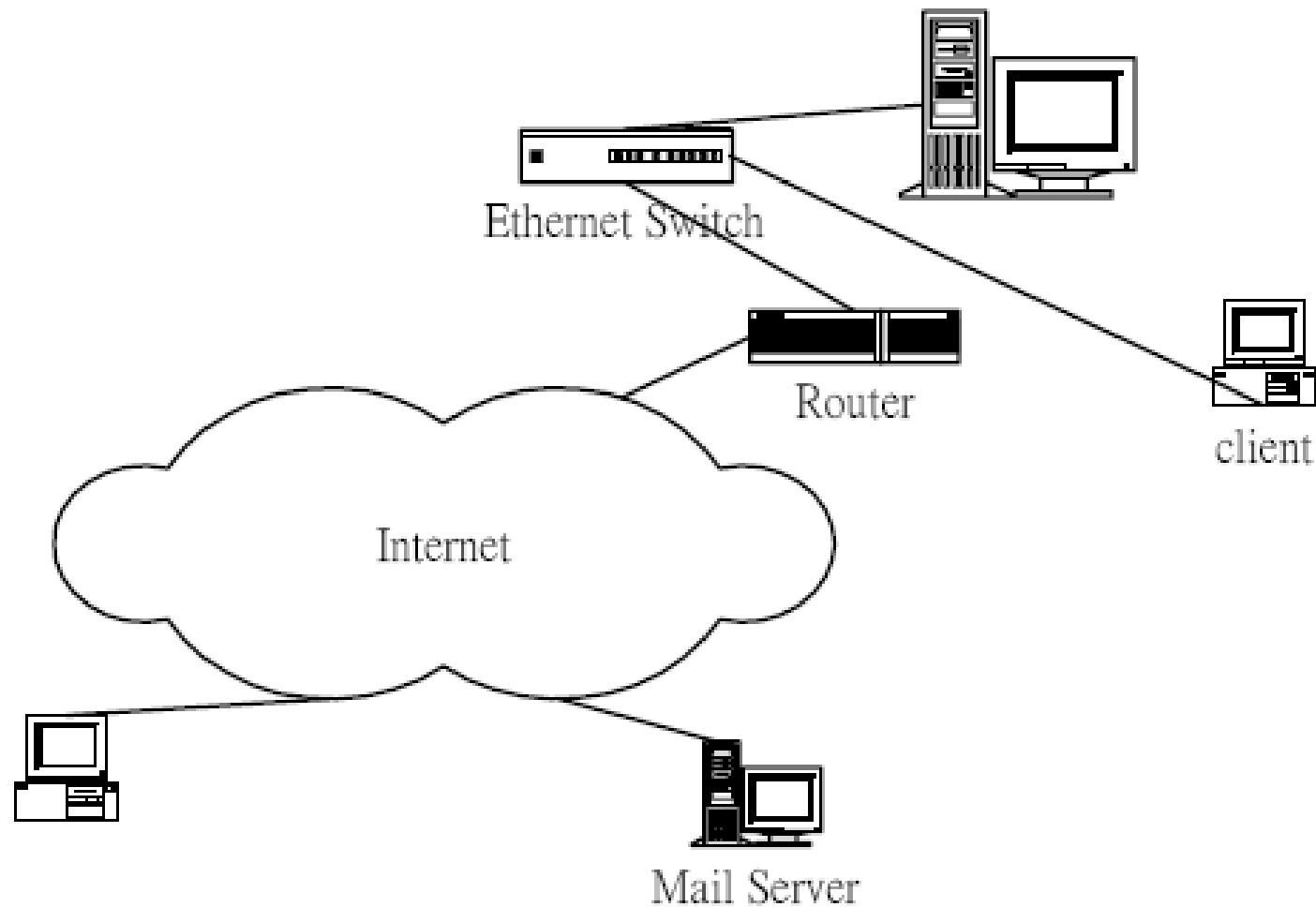
VoIP Gateway 連接 PSTN 與 Internet

- ❖ 可轉換應用層的應用稱爲 **Application Gateway**，有些則稱爲 **Application Level Gateway (ALG)**

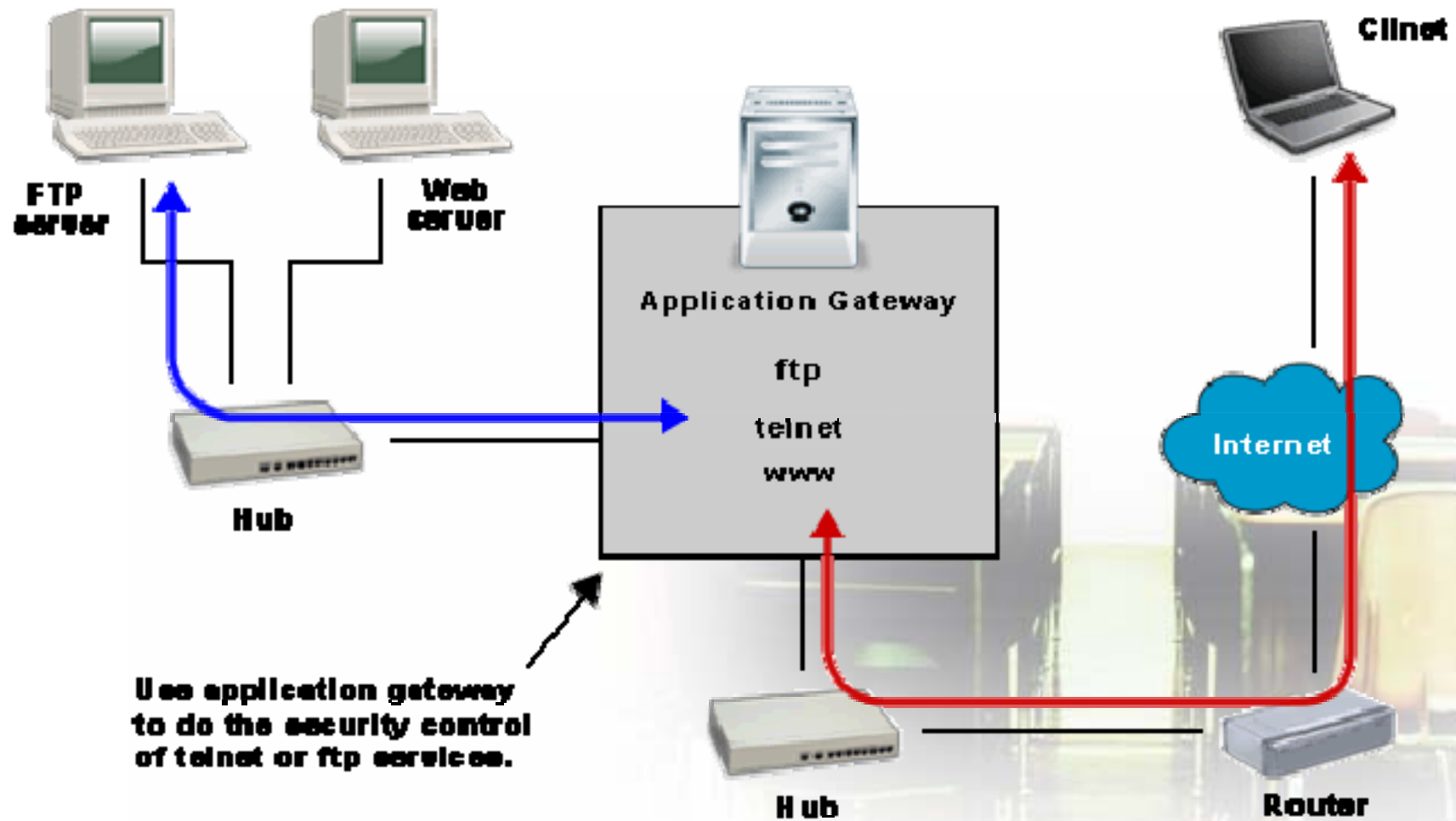


e-mail Application Gateway

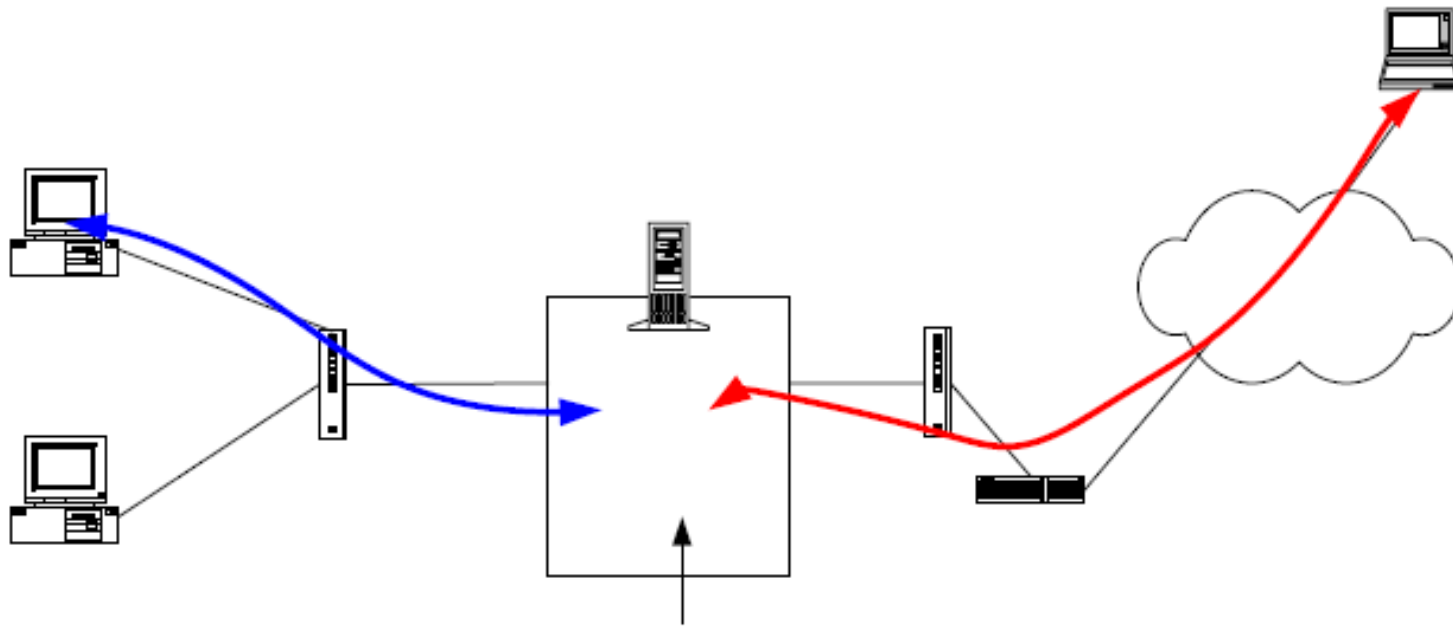
e-mail Application Gateway
/Mail Server 轉信, 過濾, 掃毒



Application Gateway代理telnet與ftp



Application Gateway代理telnet與ftp



可設計成分散伺服器的負擔平衡的設備



Default Gateway

CYUT NC

Default Gateway

- ❖ 當兩台電腦在不同網域互傳 **Packet** 時
 - 電腦會先查詢自己的Routing Table，找出往另一台電腦的路由
 - 如果另一台電腦的網域不在自己的Routing Table內，就會送往Default Gateway，由Default Gateway負責轉送 Packet
- ❖ 一般固接式的**ADSL**都是設定**ADSL ATU-R** 做為**Default Gateway**

Default Gateway的三種設定

- ❖ 由**ISP** 配發，指定**Default Gateway**，如固接專線或固接**ADSL**，用戶端必須自行設定**Default Gateway**
- ❖ 使用**DHCP** (**Dynamic Host Configuration Protocol, RFC 1541, RFC 2131**) 自動取得**Default Gateway**，如**Cable Modem**
- ❖ 使用**PPP** (**The Point-to-Point Protocol, RFC 1661**) 取得**Default Gateway**，例如：**Modem** 撥接；或撥接計時制**ADSL**使用**PPPoE** (**PPP over Ethernet**)取得 (事實上也使用**PPP**)

藉由 ip route 設定 default gateway

```
bash# ip route
192.168.2.0/24 dev eth2 proto kernel scope link src 192.168.2.1
192.168.1.0/24 dev eth1 proto kernel scope link src 192.168.1.1
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.1
192.168.251.0/24 dev eth3 proto kernel scope link src 192.168.251.251
bash#
bash#
bash# ip route add default via 192.168.2.2
bash#
bash#
bash# ip route
192.168.2.0/24 dev eth2 proto kernel scope link src 192.168.2.1
192.168.1.0/24 dev eth1 proto kernel scope link src 192.168.1.1
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.1
192.168.251.0/24 dev eth3 proto kernel scope link src 192.168.251.251
default via 192.168.2.2 dev eth2
bash#
```

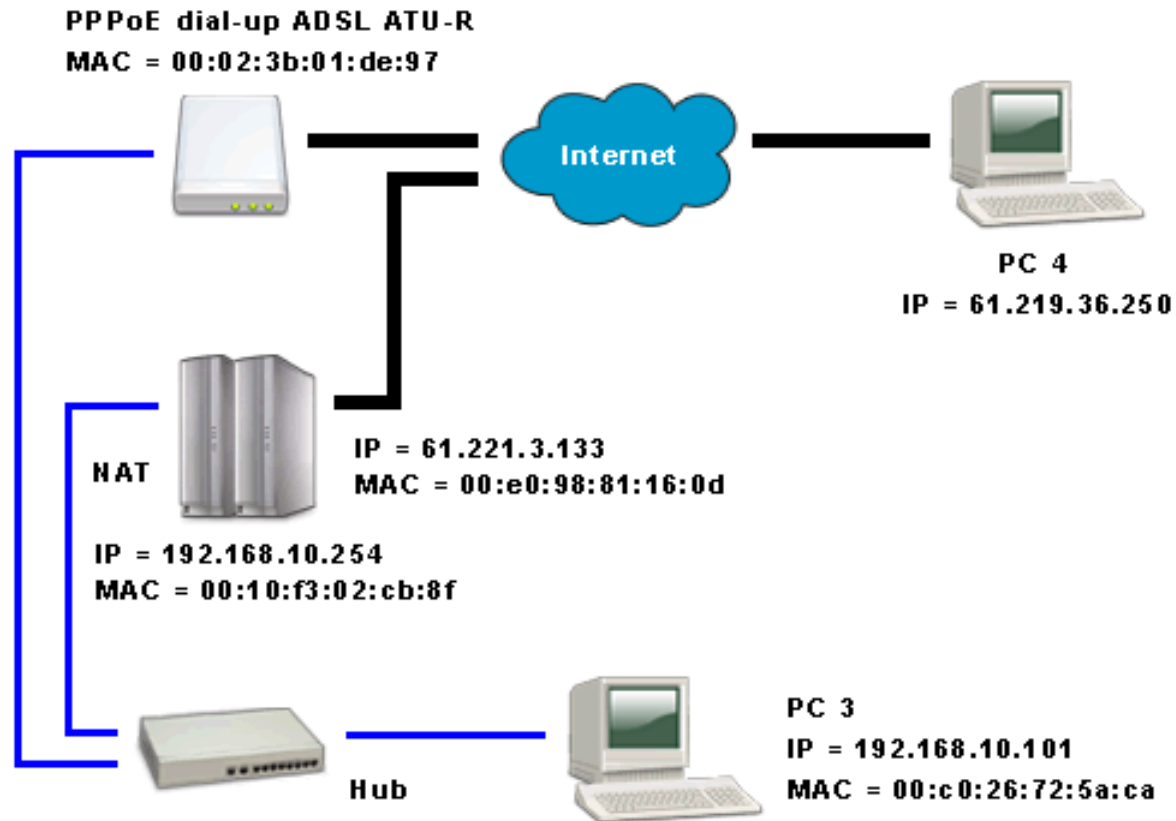
A photograph of a classroom. In the foreground, there are several rows of wooden desks and brown chairs. In the background, a large green chalkboard is mounted on the wall. The text "藉由PPPoE設定Default Gateway" is written in white on the chalkboard.

藉由PPPoE設定Default Gateway

CYUT NC

實驗架構圖

❖ PC3原Default Gateway為 192.168.10.254



PC1還沒有執行PPPoE上網撥接程式 抓取封包觀察

❖ 192.168.10.101 連線遠端的 61.219.36.250

No.	Time	Source .	Destination	Protocol	Info
4	5.340663	192.168.10.101	61.219.36.250	TCP	1186 > 80 [SYN] Seq=33035860
5	5.341331	61.225.38.42	61.219.36.250	TCP	1186 > 80 [SYN] Seq=33035860
6	5.406526	61.219.36.250	61.225.38.42	TCP	80 > 1186 [SYN, ACK] Seq=304
7	5.406746	61.219.36.250	192.168.10.101	TCP	80 > 1186 [SYN, ACK] Seq=304
8	5.406819	192.168.10.101	61.219.36.250	TCP	1186 > 80 [ACK] Seq=33035860
9	5.407075	61.225.38.42	61.219.36.250	TCP	1186 > 80 [ACK] Seq=33035860
10	5.407907	192.168.10.101	61.219.36.250	HTTP	GET / HTTP/1.1
11	5.408815	61.225.38.42	61.219.36.250	HTTP	GET / HTTP/1.1
12	5.545181	61.219.36.250	61.225.38.42	TCP	80 > 1186 [ACK] Seq=30415270
13	5.545327	61.219.36.250	192.168.10.101	TCP	80 > 1186 [ACK] Seq=30415270
14	5.587616	61.219.36.250	61.225.38.42	TCP	80 > 1186 [ACK] Seq=30415270
15	5.588772	61.219.36.250	192.168.10.101	TCP	80 > 1186 [ACK] Seq=30415270
16	5.588837	61.219.36.250	61.225.38.42	TCP	80 > 1186 [ACK] Seq=30415270
17	5.588985	61.219.36.250	192.168.10.101	TCP	80 > 1186 [ACK] Seq=30415270

Destination MAC Address is the MAC Address of 192.168.10.254. Therefore, 192.168.10.254 is the default gateway of PC1

Frame 4 (62 bytes on wire, 62 bytes captured)

Ethernet II, Src: 00:c0:26:72:5a:ca, Dst: 00:10:f3:02:cb:8f
Destination: 00:10:f3:02:cb:8f (00:10:f3:02:cb:8f)
Source: 00:c0:26:72:5a:ca (00:c0:26:72:5a:ca)
Type: IP (0x0800)

Internet Protocol, Src Addr: 192.168.10.101 (192.168.10.101), Dst Addr: 61.219.36.250 (61.219.36.250)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)

```
0000  00 10 f3 02 cb 8f 00 c0 26 72 5a ca 08 00 45 00  ..... &rZ...E.
0010  00 30 07 6e 40 00 80 06 c5 77 c0 a8 0a 65 3d db  .0.n@... .w...e.
0020  24 fa 04 a2 00 50 c4 e8 b9 20 00 00 00 00 70 02  $....P... ..p.
0030  40 00 92 41 00 00 02 04 05 b4 01 01 04 02  @..A.....
```

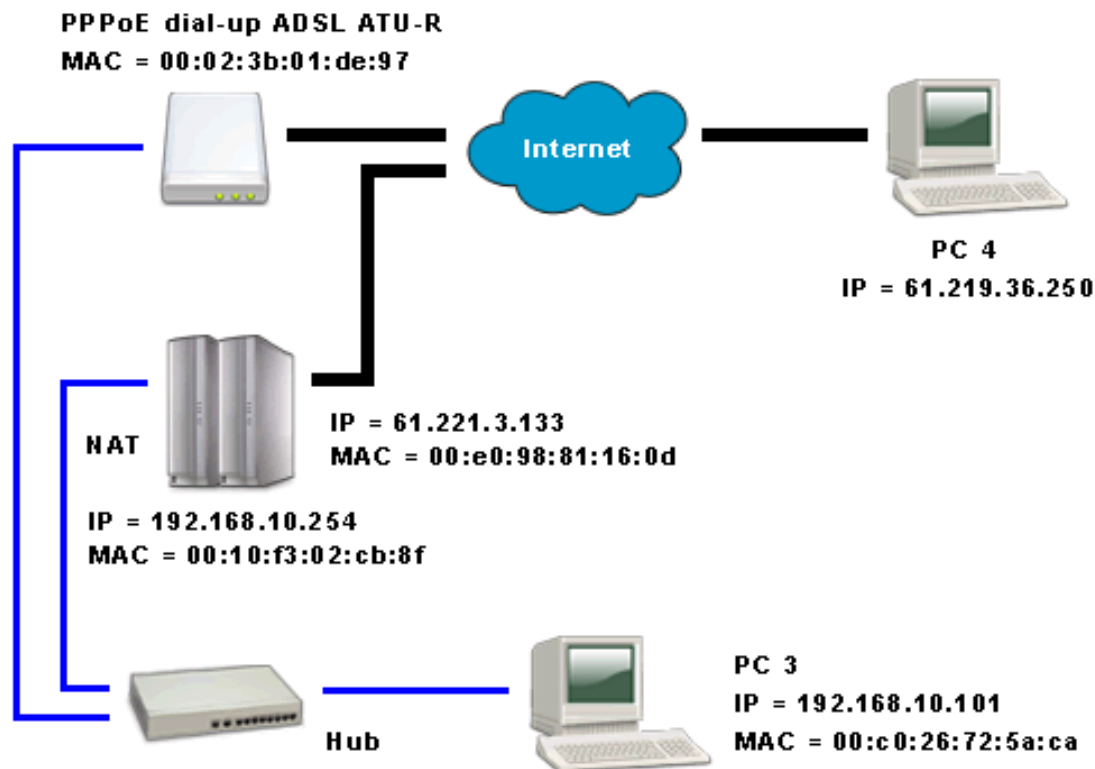
PC1還沒有執行PPPoE上網撥接程式 查詢Routing Table

❖原始的 Default Gateway為 192.168.10.254

```
C:\>route print
=====
Interface List
0x1 ..... MS TCP Loopback interface
0x2 ...44 45 53 54 77 77 ..... NTS PPPoE Adapter #1
0x1000004 ...00 c0 26 72 5a ca ..... NDIS 5.0 driver
=====
=====
Active Routes:
Network Destination        Netmask          Gateway          Interface        Metric
0.0.0.0                    0.0.0.0          192.168.10.254   192.168.10.101   1
127.0.0.0                  255.0.0.0        127.0.0.1        127.0.0.1        1
192.168.10.0               255.255.255.0    192.168.10.101   192.168.10.101   1
192.168.10.101            255.255.255.255   127.0.0.1        127.0.0.1        1
192.168.10.255            255.255.255.255   192.168.10.101   192.168.10.101   1
224.0.0.0                  224.0.0.0        192.168.10.101   192.168.10.101   1
255.255.255.255          255.255.255.255   192.168.10.101   2                1
Default Gateway:          192.168.10.254
=====
```

PC1執行PPPoE上網撥接程式後

- ❖ 經由 PPP 取得網路IP、Gateway與DNS 等設定後，並更動 Routing Table，將Default Gateway 設為由 PPP取得的 Gateway IP



PC1執行PPPoE上網撥接程式後 抓取封包觀察

上網撥接程式，同時使用Ethereal 擷取 Packet

The screenshot shows the Ethereal network capture tool interface. The main window displays a list of captured packets with the following columns: No., Time, Source, Destination, Protocol, and Info. The first packet is highlighted in blue.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	00:c0:26:72:5a:ca	ff:ff:ff:ff:ff:ff	PPPoED	Active Discovery Initiat
2	0.088106	00:02:3b:01:de:97	00:c0:26:72:5a:ca	PPPoED	Active Discovery offer (
3	0.088168	00:c0:26:72:5a:ca	00:02:3b:01:de:97	PPPoED	Active Discovery Request
4	0.155002	00:02:3b:01:de:97	00:c0:26:72:5a:ca	PPPoED	Active Discovery Session
5	0.187495	00:c0:26:72:5a:ca	00:02:3b:01:de:97	PPP LCP	PPP LCP Configuration Re
6	0.255326	00:02:3b:01:de:97	00:c0:26:72:5a:ca	PPP LCP	PPP LCP Configuration Re
7	0.256924	00:02:3b:01:de:97	00:c0:26:72:5a:ca	PPP LCP	PPP LCP Configuration Ac
8	0.312499	00:c0:26:72:5a:ca	00:02:3b:01:de:97	PPP LCP	PPP LCP Configuration Ac
9	0.312525	00:c0:26:72:5a:ca	00:02:3b:01:de:97	PPP PAP	PPP PAP Authenticate-Req
10	2.106966	00:02:3b:01:de:97	00:c0:26:72:5a:ca	PPP PAP	PPP PAP Authenticate-Ack
11	2.108572	00:02:3b:01:de:97	00:c0:26:72:5a:ca	PPP IPCP	PPP IPCP Configuration R
12	2.125081	00:c0:26:72:5a:ca	00:02:3b:01:de:97	PPP IPCP	PPP IPCP Configuration R
13	2.125105	00:c0:26:72:5a:ca	00:02:3b:01:de:97	PPP IPCP	PPP IPCP Configuration A
14	2.186911	00:02:3b:01:de:97	00:c0:26:72:5a:ca	PPP IPCP	PPP IPCP Configuration R

The details pane for the first packet shows the following information:

- Frame 1 (34 bytes on wire, 34 bytes captured)
- Ethernet II, Src: 00:c0:26:72:5a:ca, Dst: ff:ff:ff:ff:ff:ff
- PPP-over-Ethernet Discovery
 - Version: 1
 - Type: 1
 - Code: Active Discovery Initiation (PADI)
 - Session ID: 0000

The hex dump at the bottom shows the raw bytes of the packet:

```
0000 ff ff ff ff ff ff 00 c0 26 72 5a ca 88 63 11 09 ..... &rZ...C...
0010 00 00 00 0e 01 03 00 06 00 c0 26 72 5a ca 01 01 ..... ..&rZ...
0020 00 00 ..
```

PC1執行PPPoE上網撥接程式後 查詢Routing Table

Default Gateway 已改變為PPP取得之Gateway IP

```
C:\>route print
=====
Interface List
0x1 ..... MS TCP Loopback interface
0x2 ...44 45 53 54 77 77 ..... NIS PPPoE Adapter #1
0x1000004 ...00 c0 26 72 5a ca ..... NDIS 5.0 driver
=====
Active Routes:
Network Destination        Netmask          Gateway          Interface        Metric
0.0.0.0                    0.0.0.0          192.168.10.254   192.168.10.101   2
0.0.0.0                    0.0.0.0          61.225.34.50    61.225.34.50    1
61.225.34.0                255.255.255.0    61.225.34.50    61.225.34.50    1
61.225.34.50               255.255.255.255  127.0.0.1       127.0.0.1       1
61.255.255.255            255.255.255.255  61.225.34.50    61.225.34.50    1
127.0.0.0                  255.0.0.0        127.0.0.1       127.0.0.1       1
192.168.10.0               255.255.255.0    192.168.10.101  192.168.10.101  2
192.168.10.101            255.255.255.255  127.0.0.1       127.0.0.1       1
192.168.10.255            255.255.255.255  192.168.10.101  192.168.10.101  1
224.0.0.0                  224.0.0.0        61.225.34.50    61.225.34.50    1
224.0.0.0                  224.0.0.0        192.168.10.101  192.168.10.101  1
255.255.255.255           255.255.255.255  61.225.34.50    61.225.34.50    1
Default Gateway:          61.225.34.50
=====
```

Linux Router

CYUT NC

Linux Router

- ❖ 使用一般PC加上 **Linux**，經過適當的設定並結合 **Routing Protocol** 即可成爲一小型的 **Router** 使用
- ❖ 若更進一步於硬體 **Routing** 與 **Packet** 處理方面加強，如使用 **Intel IXP** 系列 **Network Processor** 或 **IBM PowerNP** 系列，這些硬體結合 **Linux** 可以設計爲一商業用之 **Router**

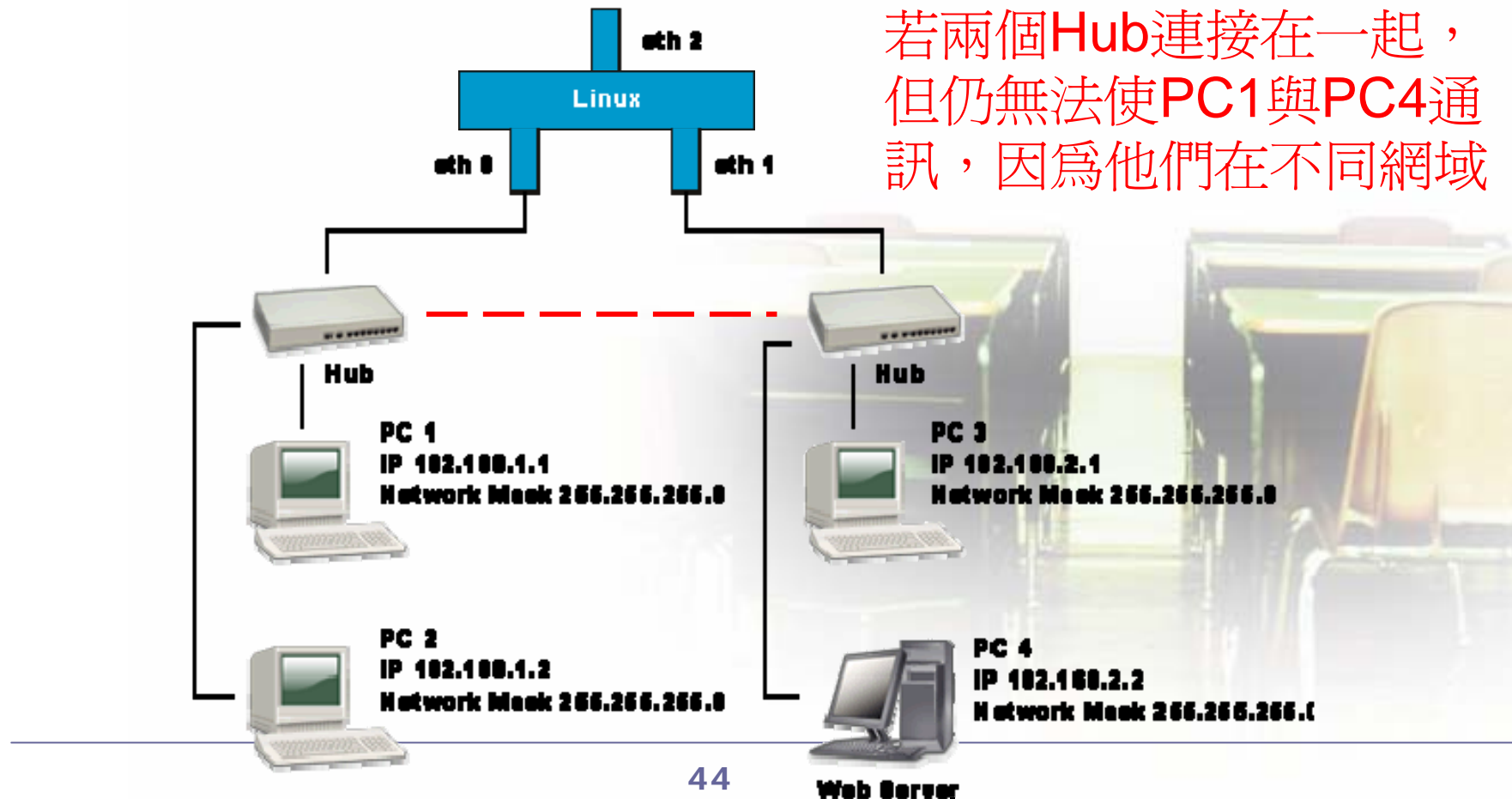
Linux router open Source

❖ 網路上有許多的**Open Source**程式碼，但是當使用時，多少需要加以修正與增加功能，甚至改變原始程式以配合硬體

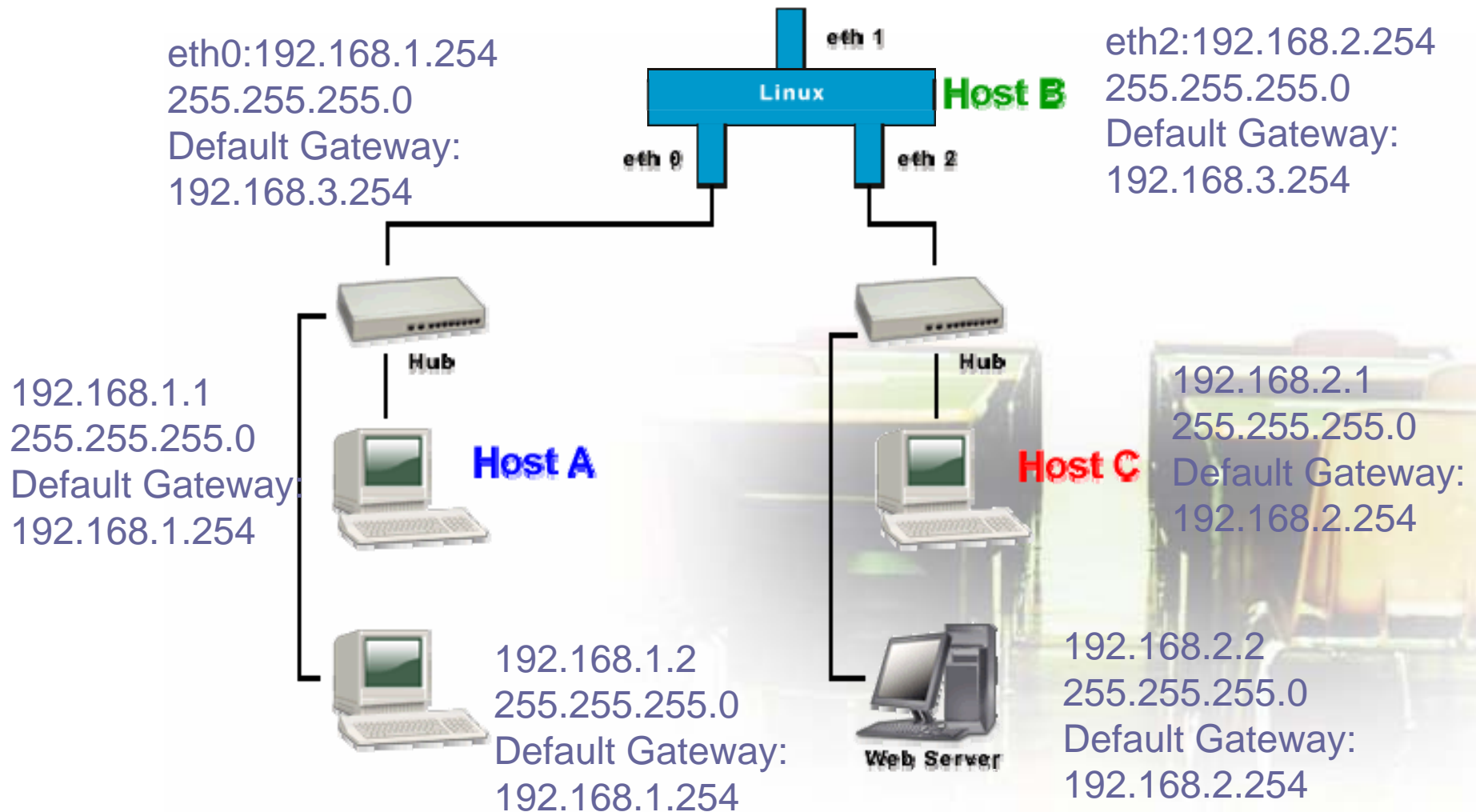
- [1] Linux Router Project, <http://www.linuxrouter.org/>
- [2] Free Cisco, <http://www.freescisco.org/>
- [3] IBM PowerNP, <http://www.ibm.com>
- [4] Intel IXP系列 Network Processor, <http://www.intel.com>
- [5] Linux Routing Protocol, <http://www.zebra.org>

Linux Router實例

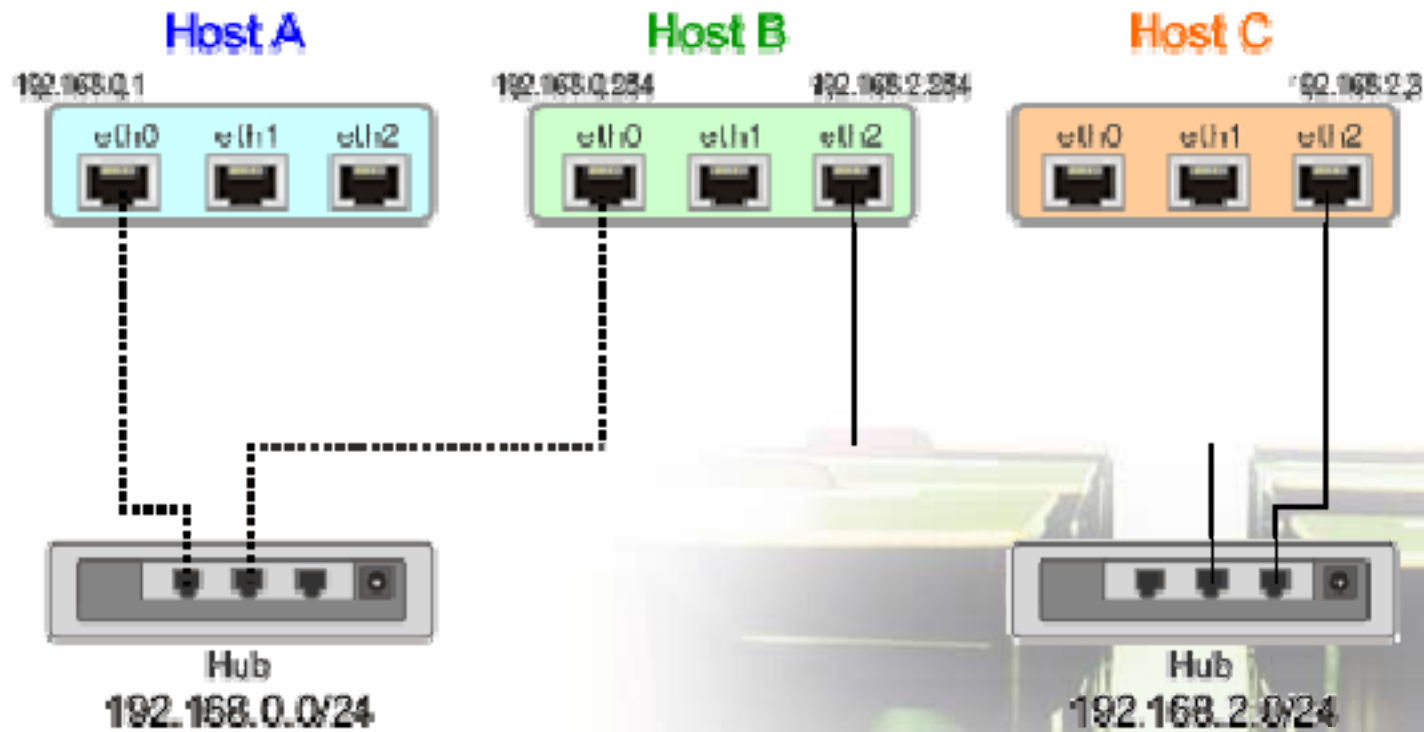
- ❖ 原本PC 1 與 PC 4 之間無法使用HTTP溝通
- ❖ 經由Linux 遠送使PC 1 與 PC 4 之間可經HTTP溝通



利用NetGuru實作



NetGuru架構圖



設定 Host A eth0

- ❖ `ifconfig eth0 192.168.0.1 netmask 255.255.255.0`
- ❖ `ip route add default via 192.168.0.254`

設定Host B eth0

❖ **ifconfig eth0 192.168.0.254 netmask 255.255.255.0**

完成此命令除了設定介面 IP 與 Network Mask 外，
ifconfig 還幫您自動設了一靜態Route 的Entry

192.168.0.0/24 dev eth0 scope link src 192.168.1.254

相當於執行了

`ip route add 192.168.1.254/24 dev eth0`

設定Host B eth2

❖ **ifconfig eth2 192.168.2.254 netmask 255.255.255.0**

完成此命令除了設定eth1介面 IP 與Network Mask 外，ifconfig 也還幫您自動設了一靜態Route 的Entry

192.168.2.0/24 dev eth2 scope link src 192.168.2.254

相當於執行了

ip route add 192.168.2.254/24 dev eth2

設定Host C eth2

- ❖ `ifconfig eth2 192.168.2.3 netmask 255.255.255.0`
- ❖ `ip route add default via 192.168.2.254`



Linux Proxy ARP

CYUT NC

概說

- ❖ **Proxy** 一詞可以用在各種場合，我們可想像其為代理人，當您想與某方完成一任務時，可以透過代理人間接幫您完成，但除了完成任務之外，此代理人還可以幫助加速、掃毒或紀錄等
- ❖ 如最常用的即是**WEB Proxy**，也就是常用的瀏覽器**Proxy** 設定中使用的**Proxy Server**。最有名的 **WEB Proxy** 即為 **squid** (<http://www.squid-cache.org/>)

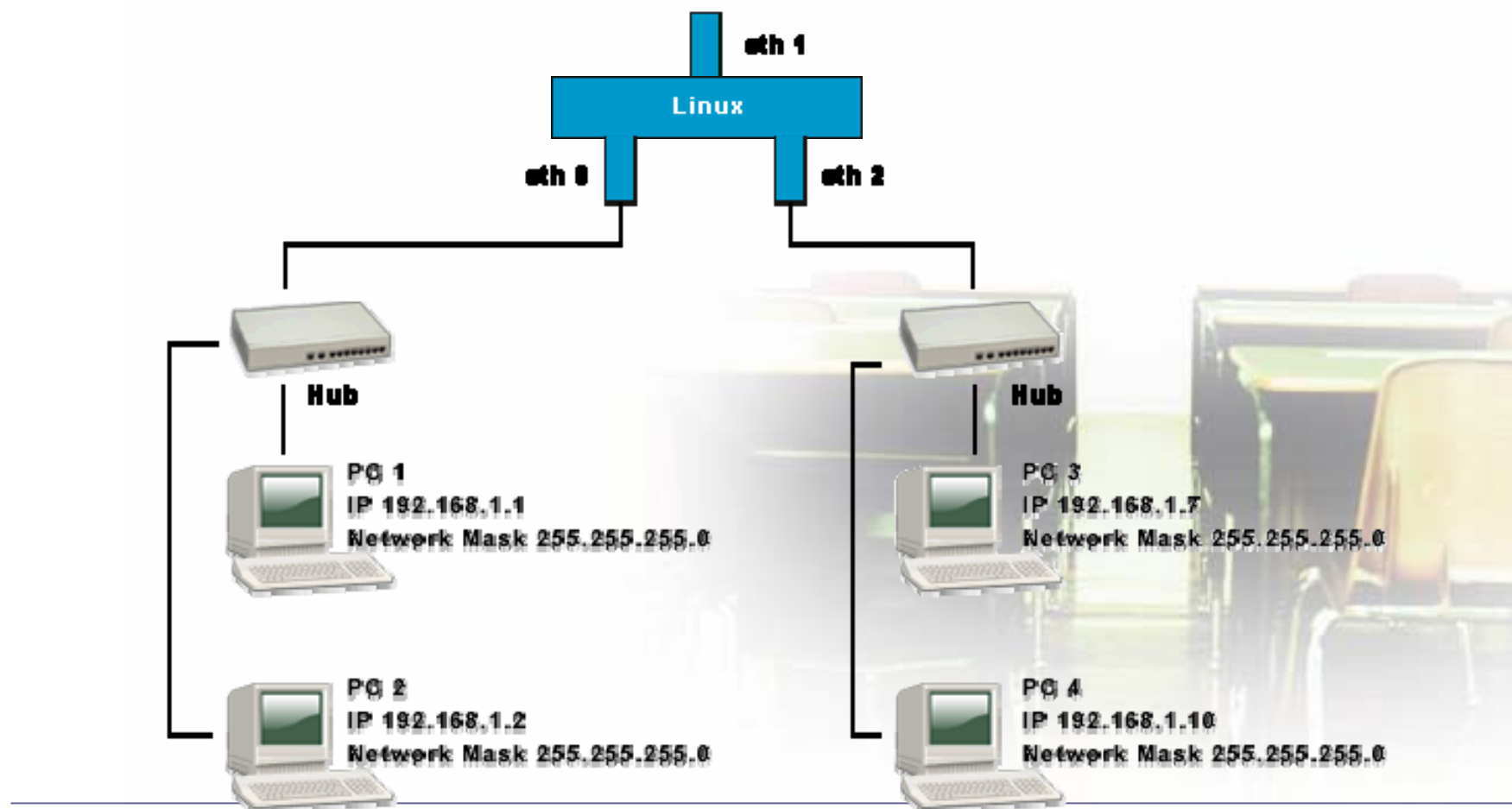
Proxy ARP

- ❖ **Proxy**是代理來源端向目的端提出請求，對於來源端而言，所面對的是**Proxy**，而目的端所回應的，也是以為來自**Proxy**，但事實的來源主機卻是在**Proxy**後面。
- ❖ **Proxy ARP**主要是針為**ARP**的協定，來源端送出**ARP**的廣播時，**Proxy ARP**必須知道(記錄)它另一端的主機情形，而代理回應，並將該**MAC Address**的資料轉送至另一端。例如：撥接伺服器便代理兩端電腦的功能**Proxy ARP**。

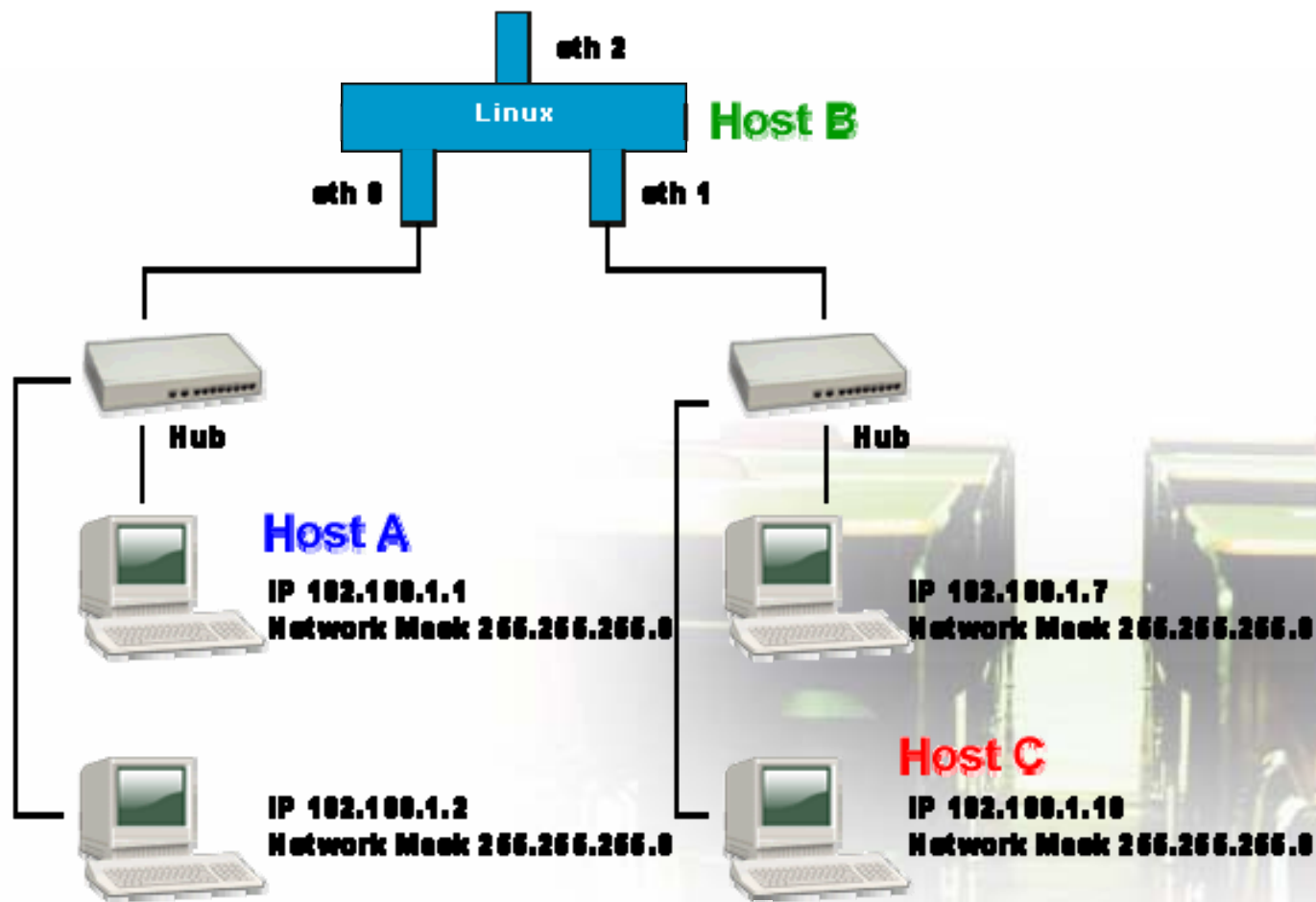
範例說明

PC 1 與 PC 4 必須透過 Proxy ARP 才能溝通

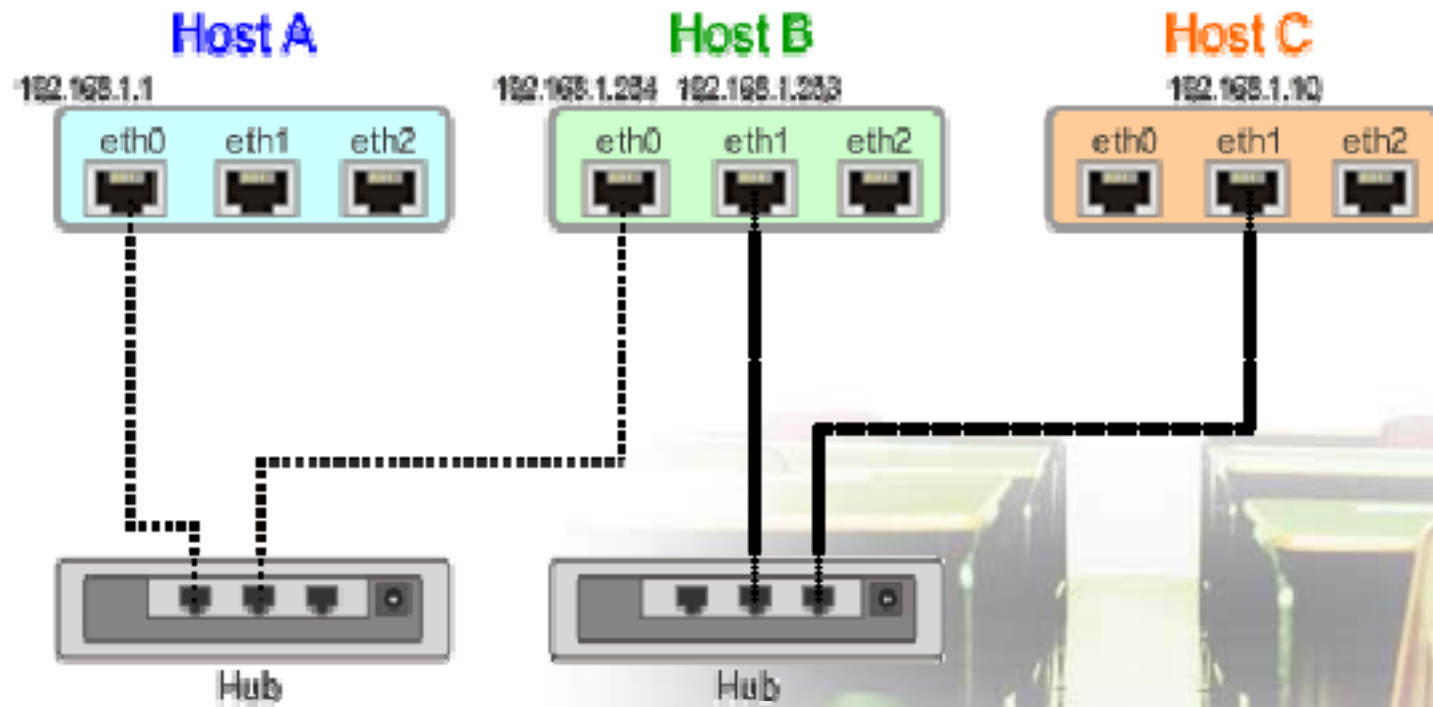
PC 4 的 MAC Address 為 Linux eth0 的 MAC Address，而對 PC 4 的 ARP Request 回應 PC 1 的 MAC Address 為 Linux eth2



利用NetGuru實作



NetGuru架構圖



NetGuru内部 Host B

```
bash# ip route
192.168.1.0/24 dev eth0 proto kernel scope link src 192.168.1.254
192.168.1.0/24 dev eth1 proto kernel scope link src 192.168.1.253
192.168.251.0/24 dev eth3 proto kernel scope link src 192.168.251.252
bash# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:30:C7:81:44:23
          inet addr:192.168.1.254  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:72 errors:0 dropped:0 overruns:0 frame:0
          TX packets:81 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:10 Base address:0xa000

eth1      Link encap:Ethernet  HWaddr 00:30:C7:81:44:22
          inet addr:192.168.1.253  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:75 errors:0 dropped:0 overruns:0 frame:0
          TX packets:75 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:11 Base address:0xc000

eth3      Link encap:Ethernet  HWaddr 00:30:C7:81:44:20
          inet addr:192.168.251.252  Bcast:192.168.251.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:9347 errors:1 dropped:0 overruns:0 frame:0
          TX packets:7329 errors:0 dropped:0 overruns:0 carrier:0
          collisions:6 txqueuelen:100
          Interrupt:12
```

NetGuru内部 Host B

下達 Proxy ARP 指令

- ❖ `arp -i eth1 -Ds 192.168.1.1 eth1 netmask 255.255.255.255 pub`
- ❖ `arp -i eth0 -Ds 192.168.1.10 eth0 netmask 255.255.255.255 pub`

```
bash# arp -an
? (192.168.1.1) at 00:30:C7:81:43:65 [ether] on eth0
? (192.168.251.254) at 00:10:F3:03:16:B3 [ether] on eth3
? (192.168.1.10) at 00:30:C7:81:41:98 [ether] on eth1
? (192.168.251.22) at 00:E0:98:81:16:0D [ether] on eth3
? (192.168.1.10) at * PERM PUP on eth0
? (192.168.1.1) at * PERM PUP on eth1
```

NetGuru内部 Host A

```
bash#  
bash# arp -an  
? (192.168.251.254) at 00:10:F3:03:16:B3 [ether] on eth3  
? (192.168.251.22) at 00:E0:98:81:16:0D [ether] on eth3  
bash# ping 192.168.1.10  
PING 192.168.1.10 (192.168.1.10) from 192.168.1.1 : 56(84) bytes of data.  
64 bytes from 192.168.1.10: icmp_seq=1 ttl=63 time=837 ms  
64 bytes from 192.168.1.10: icmp_seq=2 ttl=63 time=0.238 ms  
64 bytes from 192.168.1.10: icmp_seq=3 ttl=63 time=0.345 ms  
  
[3]+ Stopped ping 192.168.1.10  
bash# arp -an  
? (192.168.251.254) at 00:10:F3:03:16:B3 [ether] on eth3  
? (192.168.1.10) at 00:30:C7:81:44:23 [ether] on eth0  
? (192.168.251.22) at 00:E0:98:81:16:0D [ether] on eth3  
bash# ip route  
192.168.1.0/24 dev eth0 proto kernel scope link src 192.168.1.1  
192.168.251.0/24 dev eth3 proto kernel scope link src 192.168.251.251  
bash#
```

NetGuru内部 Host C

```
bash# arp -an
? (192.168.1.1) at 00:30:C7:81:44:22 [ether] on eth1
? (192.168.251.254) at 00:10:F3:03:16:B3 [ether] on eth3
? (192.168.251.22) at 00:E0:98:81:16:0D [ether] on eth3
bash# ping 192.168.1.253
PING 192.168.1.253 (192.168.1.253) from 192.168.1.10 : 56(84) bytes of data.
64 bytes from 192.168.1.253: icmp_seq=1 ttl=64 time=0.322 ms
64 bytes from 192.168.1.253: icmp_seq=2 ttl=64 time=0.158 ms

[7]+  Stopped                  ping 192.168.1.253
bash# arp -an
? (192.168.1.253) at 00:30:C7:81:44:22 [ether] on eth1
? (192.168.1.1) at 00:30:C7:81:44:22 [ether] on eth1
? (192.168.251.254) at 00:10:F3:03:16:B3 [ether] on eth3
? (192.168.251.22) at 00:E0:98:81:16:0D [ether] on eth3
bash#
```

NetGuru内部 Host B 抓取eth0的封包

The screenshot shows the Ethereal network capture tool interface. The main window displays a list of captured packets. Packet 3 is selected, showing it is an ICMP Echo (ping) request from 192.168.1.10 to 192.168.1.1. Below the list, the details for this packet are shown, including the Ethernet II header, Internet Protocol header, and Internet Control Message Protocol header. The packet data is displayed in hexadecimal and ASCII format.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	00:30:c7:81:44:23	ff:ff:ff:ff:ff:ff	ARP	Who has 192.168.1.1? Tell 192.168.1.254
2	0.000122	00:30:c7:81:43:65	00:30:c7:81:44:23	ARP	192.168.1.1 is at 00:30:c7:81:43:65
3	0.000136	192.168.1.10	192.168.1.1	ICMP	Echo (ping) request
4	0.000316	00:30:c7:81:43:65	ff:ff:ff:ff:ff:ff	ARP	Who has 192.168.1.10? Tell 192.168.1.1
5	0.009818	00:30:c7:81:44:23	00:30:c7:81:43:65	ARP	192.168.1.10 is at 00:30:c7:81:44:23
6	0.009884	192.168.1.1	192.168.1.10	ICMP	Echo (ping) reply
7	0.972128	192.168.1.10	192.168.1.1	ICMP	Echo (ping) request
8	0.972230	192.168.1.1	192.168.1.10	ICMP	Echo (ping) reply
9	1.972171	192.168.1.10	192.168.1.1	ICMP	Echo (ping) request
10	1.972345	192.168.1.1	192.168.1.10	ICMP	Echo (ping) reply
11	2.972244	192.168.1.10	192.168.1.1	ICMP	Echo (ping) request
12	2.972341	192.168.1.1	192.168.1.10	ICMP	Echo (ping) reply
13	3.972281	192.168.1.10	192.168.1.1	ICMP	Echo (ping) request
14	3.972463	192.168.1.1	192.168.1.10	ICMP	Echo (ping) reply

Frame 3 (98 bytes on wire, 98 bytes captured)
Ethernet II, Src: 00:30:c7:81:44:23, Dst: 00:30:c7:81:43:65
Internet Protocol, Src Addr: 192.168.1.10 (192.168.1.10), Dst Addr: 192.168.1.1 (192.168.1.1)
Internet Control Message Protocol

```
0000  00 30 c7 81 43 65 00 30 c7 81 44 23 08 00 45 00  .0..Ce.0 ..D#..E.  
0010  00 54 00 00 40 00 3f 01 b8 4d c0 a8 01 0a c0 a8  .T..@.?. .M.....  
0020  01 01 08 00 15 6e 86 05 01 00 16 a5 02 40 4f a4  .....n.. .....@.  
0030  08 00 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15  .....  
0040  16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25  ..... !"#%$
```

Filter: / Reset Apply File: <capture> Drops: 0

NetGuru内部 Host B 抓取eth1的封包

The screenshot shows the Wireshark interface with a capture of network traffic. The main pane displays a list of 14 captured packets. Packet 3 is selected, showing it is an ICMP Echo (ping) request from 192.168.1.10 to 192.168.1.1. The packet details pane below shows the structure: Ethernet II, Internet Protocol, and Internet Control Message Protocol. The packet bytes pane at the bottom shows the raw hex and ASCII data.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	00:30:c7:81:41:98	ff:ff:ff:ff:ff:ff	ARP	Who has 192.168.1.1? Tell 192.168.1.10
2	0.031423	00:30:c7:81:44:22	00:30:c7:81:41:98	ARP	192.168.1.1 is at 00:30:c7:81:44:22
3	0.031521	192.168.1.10	192.168.1.1	ICMP	Echo (ping) request
4	0.041538	192.168.1.1	192.168.1.10	ICMP	Echo (ping) reply
5	1.003680	192.168.1.10	192.168.1.1	ICMP	Echo (ping) request
6	1.003865	192.168.1.1	192.168.1.10	ICMP	Echo (ping) reply
7	2.003708	192.168.1.10	192.168.1.1	ICMP	Echo (ping) request
8	2.003984	192.168.1.1	192.168.1.10	ICMP	Echo (ping) reply
9	3.003779	192.168.1.10	192.168.1.1	ICMP	Echo (ping) request
10	3.003977	192.168.1.1	192.168.1.10	ICMP	Echo (ping) reply
11	4.003818	192.168.1.10	192.168.1.1	ICMP	Echo (ping) request
12	4.004100	192.168.1.1	192.168.1.10	ICMP	Echo (ping) reply
13	5.041447	00:30:c7:81:44:22	00:30:c7:81:41:98	ARP	Who has 192.168.1.10? Tell 192.168.1.253
14	5.041545	00:30:c7:81:41:98	00:30:c7:81:44:22	ARP	192.168.1.10 is at 00:30:c7:81:41:98

Frame 3 (98 bytes on wire, 98 bytes captured)
Ethernet II, Src: 00:30:c7:81:41:98, Dst: 00:30:c7:81:44:22
Internet Protocol, Src Addr: 192.168.1.10 (192.168.1.10), Dst Addr: 192.168.1.1 (192.168.1.1)
Internet Control Message Protocol

```
0000  00 30 c7 81 44 22 00 30 c7 81 41 98 08 00 45 00  .0..D".0 ..A...E.  
0010  00 54 00 00 40 00 40 01 b7 4d c0 a8 01 0a c0 a8  .T..@.@. .M.....  
0020  01 01 08 00 15 6e 86 05 01 00 16 a5 02 40 4f a4  .....n.. .....@.  
0030  08 00 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15  .....  
0040  16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25  ..... ..!"#$%
```

實驗導引

實驗 5.1 Routing設定與TTL觀察

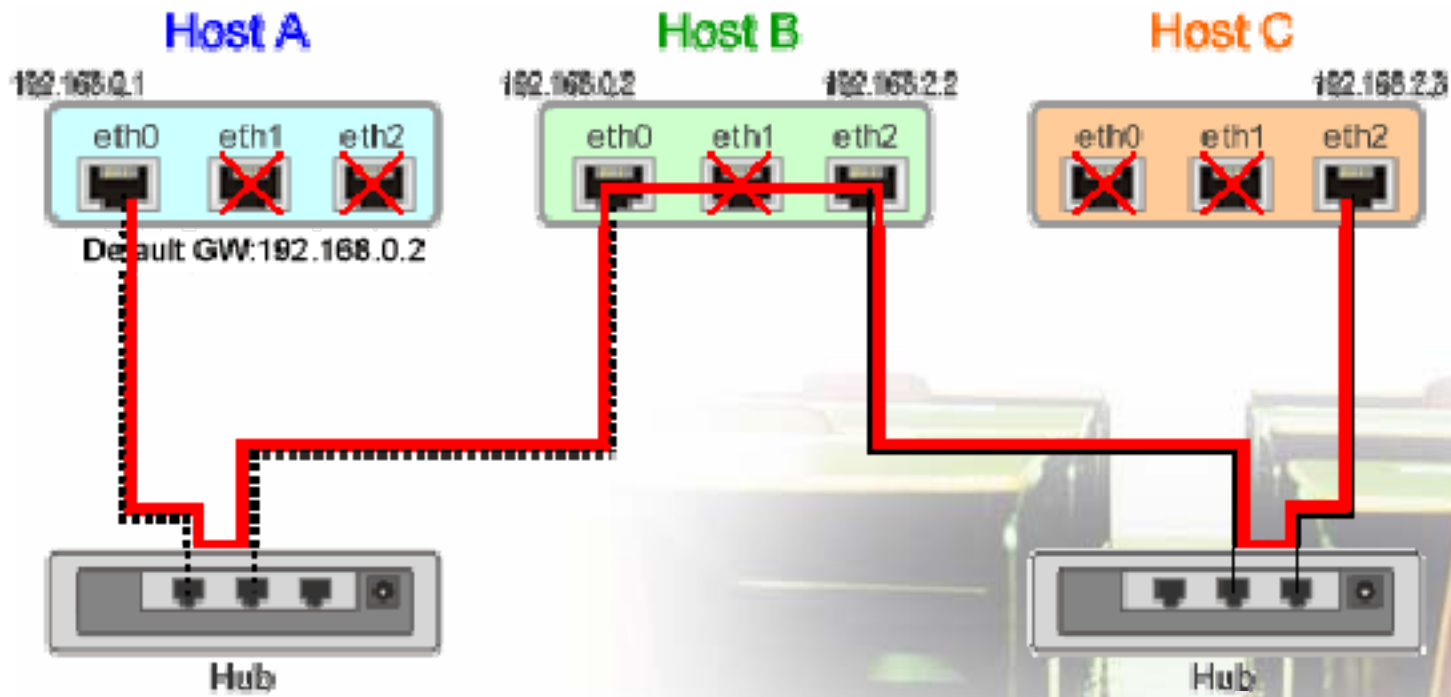
CYUT NC

實驗 5.1 Routing設定與TTL觀察

實驗目的

- ❖ 了解如何設定**Gateway**
- ❖ 了解**routing**相關概念
- ❖ 了解**default route**的意義
- ❖ 了解**TTL**相關概念
- ❖ 觀察**route**與**TTL**的變化

第一次routing架構圖



Step 1: 實驗環境設定

❖ 停用與實驗無關的網卡

❖ **Host A:**

- `ifconfig eth1 down`
- `ifconfig eth2 down`

❖ **Host B:**

- `ifconfig eth1 down`

❖ **Host C:**

- `ifconfig eth0 down`
- `ifconfig eth1 down`



Step 2: 設定 Host B 為 Host A 的 default gateway

❖ Host A:

```
Chinese RXVT (zh_TW.Big5)
bash# ip route
192.168.2.0/24 dev eth2 proto kernel scope link src 192.168.2.1
192.168.1.0/24 dev eth1 proto kernel scope link src 192.168.1.1
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.1
192.168.251.0/24 dev eth3 proto kernel scope link src 192.168.251.251
bash# ip route add default via 192.168.0.2
bash# ip route
192.168.2.0/24 dev eth2 proto kernel scope link src 192.168.2.1
192.168.1.0/24 dev eth1 proto kernel scope link src 192.168.1.1
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.1
192.168.251.0/24 dev eth3 proto kernel scope link src 192.168.251.251
default via 192.168.0.2 dev eth0
```

- ❖ Host B: 設定 IP 轉送功能
 - echo "1" > /proc/sys/net/ipv4/ip_forward

❖ 復習：何謂 default gateway

Step 3: 連線測試

❖ Host B:

- 開啓Ethereal，interface選eth0，以觀察連線測試之封包

❖ Host C:

- 開啓Ethereal，interface選eth2，以觀察連線測試之封包

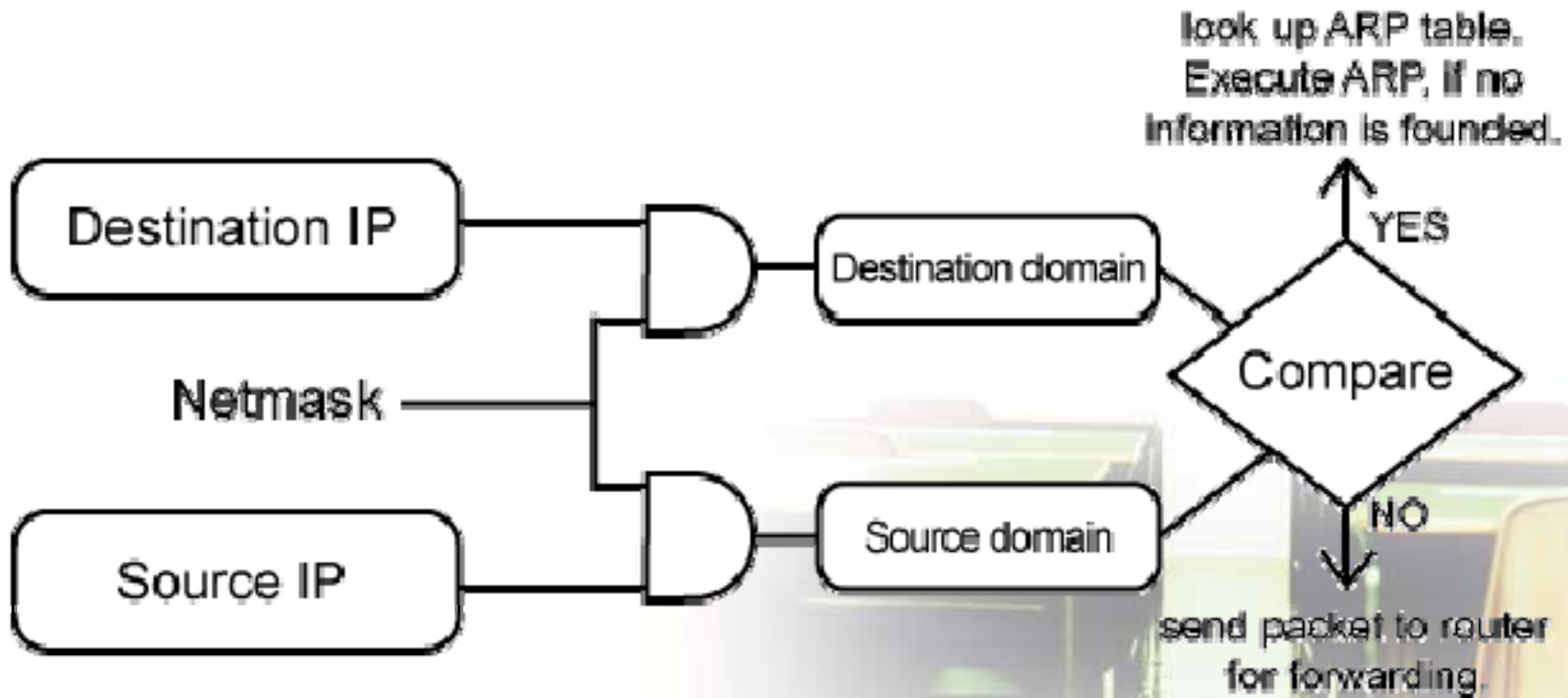
❖ Host A:

- ping 192.168.2.3
- 觀察是否有回應，並探討其原因
- 如無法連線，該如何進行問題排除

❖ 問題與討論：

- 比較Host B與Host C的Ethereal所抓到的ICMP封包的

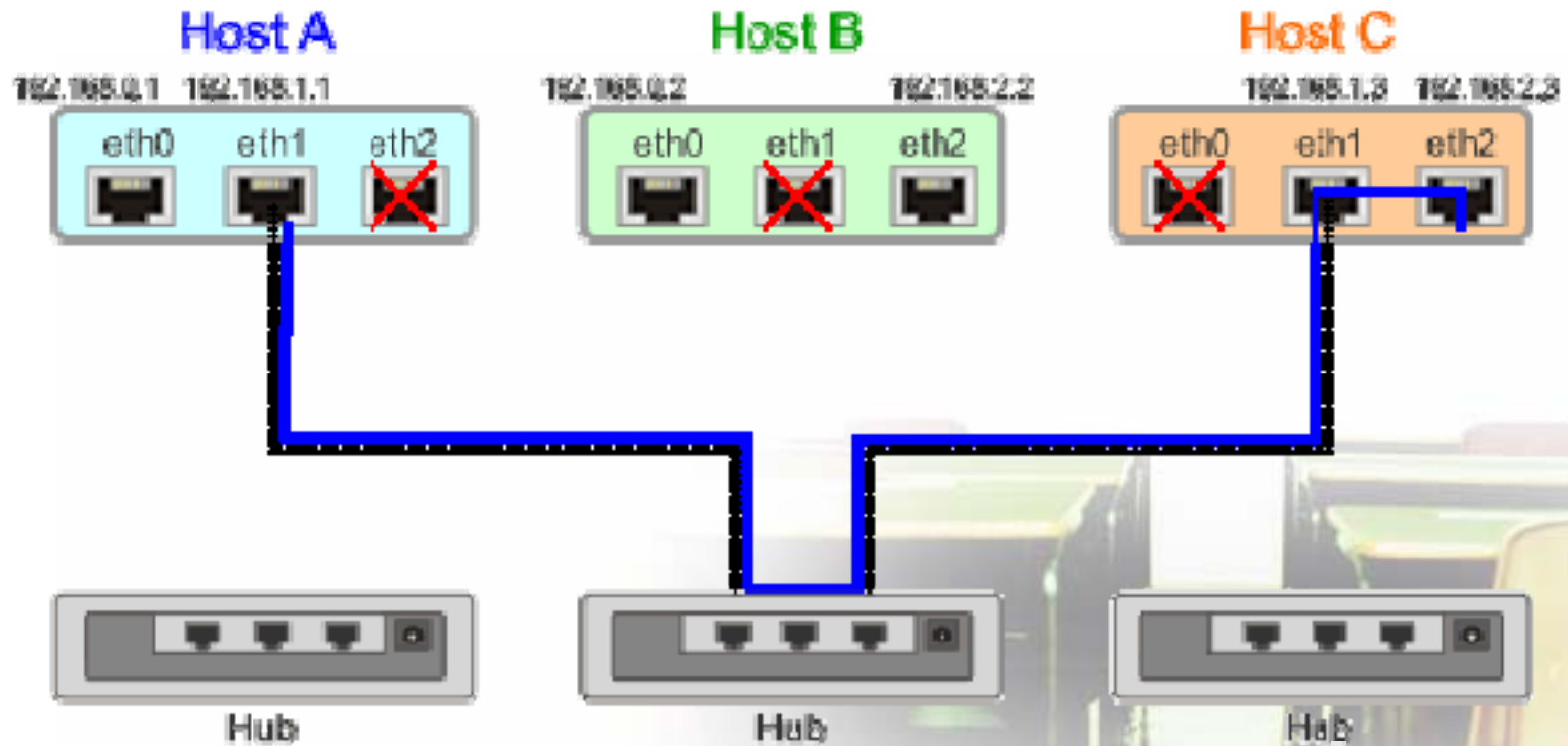
復習



Step 4: 問題排除

- ❖ 設定Host C回到192.168.0.0/24的路徑
 - `ip route add default via 192.168.2.2`
- ❖ 上面做法是使**ICMP request**和**reply**的路徑相同，然而**ICMP request**和**reply**的路徑不一定相同，如接下來的步驟所示

第二次routing架構圖



Step 5: 實驗環境設定

❖ Host A的eth1回復預設值:

- `ifconfig eth1 192.168.1.1 netmask 255.255.255.0`

❖ Host B:

- 沿用之前步驟的設定值不需更改

❖ Host C的eth1回復預設值:

- `ifconfig eth1 192.168.1.3 netmask 255.255.255.0`

Step 6: 設定路徑

❖ 設定Host C往192.168.0.0/24的路徑

❖ Host C:

- ip route add 192.168.0.0/24 via 192.168.1.1
- 使Host C藉由192.168.1.1直接到達Host A(不經Host B轉送封包)

Step 7: 連線測試

❖ Host C:

- 開啓Ethereal，interface選eth2，以觀察連線測試之封包

❖ Host A:

- 開啓Ethereal，interface選eth1，以觀察連線測試之封包

❖ Host A:

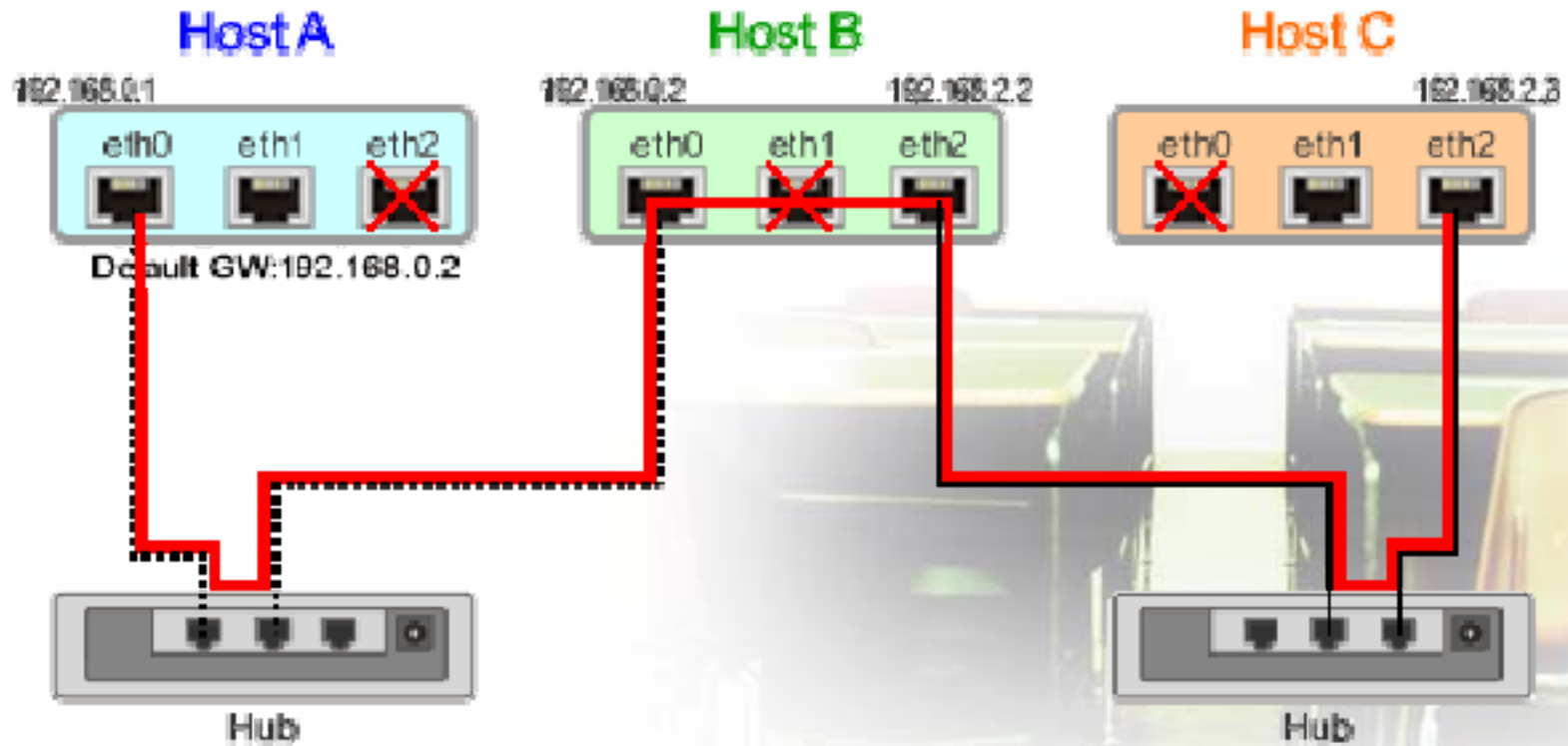
- ping 192.168.2.3，觀察TTL值並探討其原因
- traceroute 192.168.2.3

問題與討論

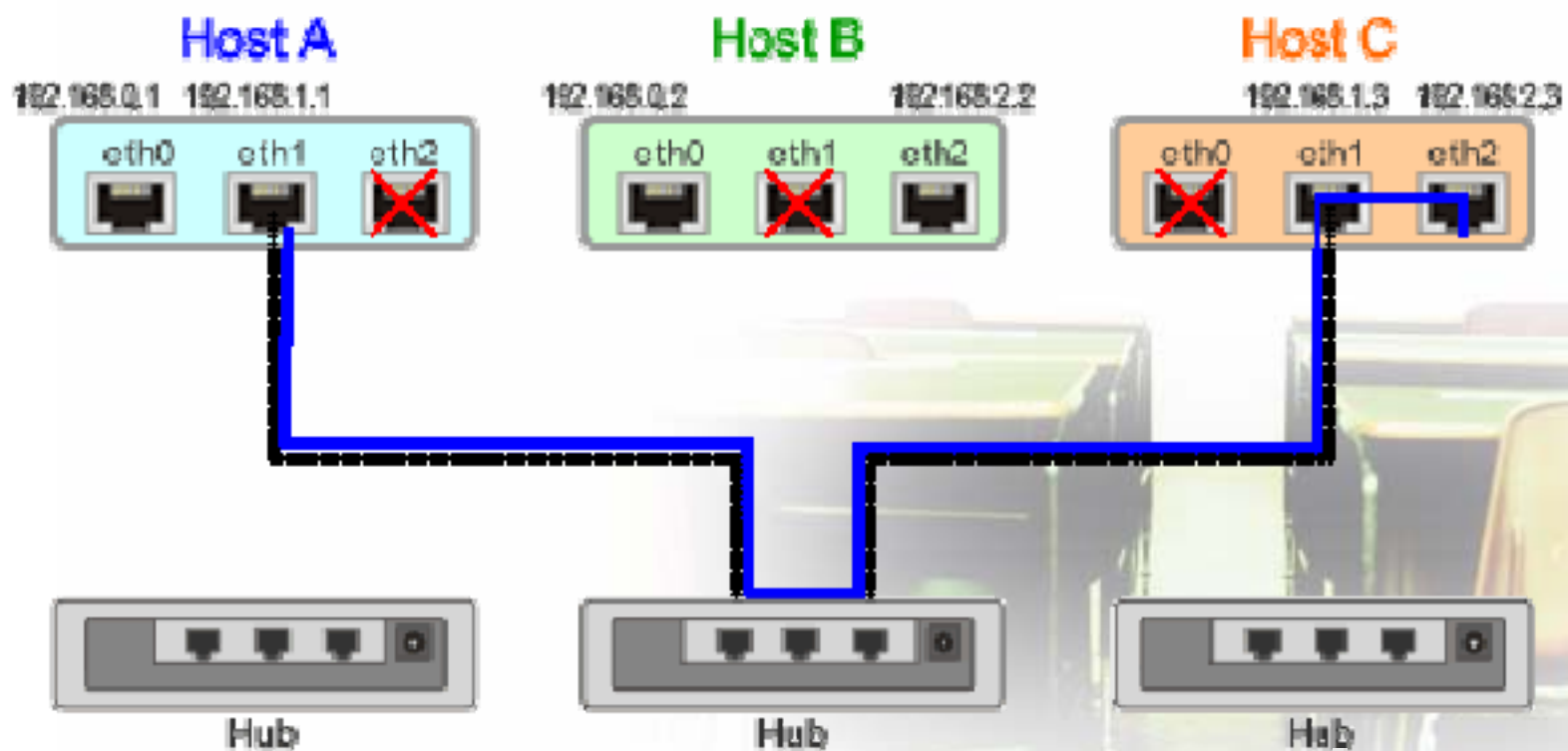
- ❖ 為何在Host C Ethereal所觀察到的ICMP封包其TTL值為63?
- ❖ 為何在Host A Ethereal所觀察到的ICMP封包其TTL值為64?
- ❖ 為何ping 192.168.2.3的TTL值為64? ping回復的TTL值是由哪一主機所決定?
- ❖ 為何Step 4 (p.69)所設定Host C到192.168.0.0/24的路徑(default route)在上步驟(Step 7)的連線測試沒有生效?
- ❖ 發出ICMP request封包的interface是否必須與接收ICMP reply的interface相同?

分析封包行經路徑

❖ ICMP request



❖ ICMP reply



Step 8: 更改TTL預設值

❖ Host C:

- `echo "260"> /proc/sys/net/ipv4/ip_default_ttl`
- `ping 192.168.0.1`
- 討論其TTL=64的意義，並分析封包行經路徑

❖ Host A:

- `ping 192.168.2.3`
- 討論其TTL=4的意義，並分析封包行經路徑

Hint: TTL的極大值為255