鳥哥的 Linux 私房菜

為取得較佳瀏覽結果，請愛用 firefox 瀏覽本網頁

第五章、 Linux 常用網路指令

切換解析度為 800x600

最近更新日期：2011/07/18

Linux 的網路功能相當的強悍，一時之間我們也無法完全的介紹所有的網路指令，這個章節主要的目的在介紹一些常見的網路指令而已。 至於每個指令的詳細用途將在後續伺服器架設時，依照指令的相關性來進行說明。當然，在這個章節的主要目的是在於將所有的指令彙整在一起， 比較容易瞭解啦！這一章還有個相當重要的重點，那就是封包擷取的指令。若不熟悉也沒關係，先放著，全部讀完後再回來這一章仔細練習啊！

5.1 網路參數設定使用的指令

任何時刻如果你想要做好你的網路參數設定，包括 IP 參數、路由參數與無線網路等等，就得要瞭解底下這些相關的指令才行！其中以 ifconfig 及 route 這兩支指令算是較重要的喔！ ^_^！當然，比較新鮮的作法，可以使用 ip 這個彙整的指令來設定 IP 參數啦！

- **ifconfig** ：查詢、設定網路卡與 IP 網域等相關參數；
- **ifup, ifdown**：這兩個檔案是 script，透過更簡單的方式來啟動網路介面；
- **route** ：查詢、設定路由表 (route table)
- **ip** ：複合式的指令， 可以直接修改上述提到的功能；

5.1.1 手動/自動設定與啟動/關閉 IP 參數： ifconfig, ifup, ifdown

這三個指令的用途都是在啟動網路介面，不過， ifup 與 ifdown 僅能就 /etc/sysconfig/network-scripts 內的 ifcfg-ethX (X 為數字) 進行啟動或關閉的動作，並不能直接修改網路參數，除非手動調整 ifcfg-ethX 檔案才行。至於 ifconfig 則可以直接手動給予某個介面 IP 或調整其網路參數！底下我們就分別來談一談！

- **ifconfig**

ifconfig 主要是可以手動的啟動、觀察與修改網路介面的相關參數，可以修改的參數很多啊，包括 IP 參數以及 MTU 等等都可以修改，他的語法如下：

```
[root@www ~]# ifconfig {interface} {up|down}   <== 觀察與啟動介面
[root@www ~]# ifconfig interface {options}      <== 設定與修改介面
選項與參數：
interface：網路卡介面代號，包括 eth0, eth1, ppp0 等等
options ：可以接的參數，包括如下：
  up, down ：啟動 (up) 或關閉 (down) 該網路介面(不涉及任何參數)
  mtu     ：可以設定不同的 MTU 數值，例如 mtu 1500 (單位為 byte)
  netmask ：就是子遮罩網路；
  broadcast：就是廣播位址啊！

# 範例一：觀察所有的網路介面(直接輸入 ifconfig)
[root@www ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:71:85:BD
          inet addr:192.168.1.100  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe71:85bd/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2555 errors:0 dropped:0 overruns:0 frame:0
          TX packets:70 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:239892 (234.2 KiB)  TX bytes:11153 (10.8 KiB)
```

一般來說，直接輸入 ifconfig 就會列出目前已經被啟動的卡，不論這個卡是否有給予 IP，都會被顯示出來。而如果是輸入 ifconfig eth0，則僅會秀出這張介面的相關資料， 而不管該介面是否有啟動。所以如果你想要知道某張網路卡的 Hardware Address，直接輸入『 ifconfig "網路介面代號" 』即可喔！ ^_^！至於上表出現的各項資料是這樣的(資料排列由上而下、由左而右)：

- **eth0**：就是網路卡的代號，也有 lo 這個 loopback ；
- **HWaddr**：就是網路卡的硬體位址，俗稱的 MAC 是也；
- **inet addr**：IPv4 的 IP 位址，後續的 Bcast, Mask 分別代表的是 Broadcast 與 netmask 喔！
- **inet6 addr**：是 IPv6 的版本的 IP ，我們沒有使用，所以略過；

- **MTU**：就是第二章談到的 MTU 啊！
- **RX**：那一行代表的是網路由啟動到目前為止的封包接收情況， packets 代表封包數、errors 代表封包發生錯誤的數量、 dropped 代表封包由於有問題而遭丟棄的數量等等
- **TX**：與 RX 相反，為網路由啟動到目前為止的傳送情況；
- **collisions**：代表封包碰撞的情況，如果發生太多次， 表示你的網路狀況不太好；
- **txqueuelen**：代表用來傳輸資料的緩衝區的儲存長度；
- **RX bytes, TX bytes**：總接收、傳送的位元組總量

透過觀察上述的資料，大致上可以瞭解到你的網路情況，尤其是那個 RX, TX 內的 error 數量， 以及是否發生嚴重的 collision 情況，都是需要注意的喔！ ^_^

```
# 範例二：暫時修改網路介面，給予 eth0 一個 192.168.100.100/24 的參數
[root@www ~]# ifconfig eth0 192.168.100.100
# 如果不加任何其他參數，則系統會依照該 IP 所在的 class 範圍，自動的計算出
# netmask 以及 network, broadcast 等 IP 參數，若想改其他參數則：

[root@www ~]# ifconfig eth0 192.168.100.100 \
> netmask 255.255.255.128 mtu 8000
# 設定不同參數的網路介面，同時設定 MTU 的數值！

[root@www ~]# ifconfig eth0 mtu 9000
# 僅修改該介面的 MTU 數值，其他的保持不動！

[root@www ~]# ifconfig eth0:0 192.168.50.50
# 仔細看那個介面是 eth0:0 喔！那就是在該實體網卡上，再模擬一個網路介面，
# 亦即是在一張網路卡上面設定多個 IP 的意思啦！

[root@www ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:71:85:BD
          inet addr:192.168.100.100  Bcast:192.168.100.127  Mask:255.255.255.128
          inet6 addr: fe80::a00:27ff:fe71:85bd/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:9000  Metric:1
          RX packets:2555 errors:0 dropped:0 overruns:0 frame:0
          TX packets:70 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:239892 (234.2 KiB)  TX bytes:11153 (10.8 KiB)

eth0:0    Link encap:Ethernet  HWaddr 08:00:27:71:85:BD
          inet addr:192.168.50.50  Bcast:192.168.50.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:9000  Metric:1
# 仔細看，是否與硬體有關的資訊都相同啊！沒錯！因為是同一張網卡嘛！
# 那如果想要將剛剛建立的那張 eth0:0 關閉就好，不影響原有的 eth0 呢？

[root@www ~]# ifconfig eth0:0 down
# 關掉 eth0:0 這個介面。那如果想用預設值啟動 eth1：『ifconfig eth1 up』即可達成

# 範例三：將手動的處理全部取消，使用原有的設定值重建網路參數：
[root@www ~]# /etc/init.d/network restart
# 剛剛設定的資料全部失效，會以 ifcfg-ethX 的設定為主！
```

使用 ifconfig 可以暫時手動來設定或修改某個介面卡的相關功能，並且也可以透過 eth0:0 這種虛擬的網路介面來設定好一張網路卡上面的多個 IP 喔！手動的方式真是簡單啊！並且設定錯誤也不打緊，因為我們可以利用 /etc/init.d/network restart 來重新啟動整個網路介面，那麼之前手動的設定資料會全部都失效喔！另外， 要啟動某個網路介面，但又不讓他具有 IP 參數時，直接給他 ifconfig eth0 up 即可！ 這個動作經常在無線網卡當中會進行，因為我們必須要啟動無線網卡讓他去偵測 AP 存在與否啊！

- **ifup, ifdown**

即時的手動修改一些網路介面參數，可以利用 ifconfig 來達成，如果是要直接以設定檔， 亦即是在 /etc/sysconfig/network-scripts 裡面的 ifcfg-ethx 等檔案的設定參數來啟動的話， 那就得要透過 ifdown 或 ifup 來達成了。

```
[root@www ~]# ifup   {interface}
[root@www ~]# ifdown {interface}

[root@www ~]# ifup eth0
```

ifup 與 ifdown 真是太簡單了！這兩支程式其實是 script 而已，他會直接到 /etc/sysconfig/network-scripts 目錄下搜尋對應的設定檔，例如 ifup **eth0** 時，他會找出 ifcfg-**eth0** 這個檔案的內容，然後來加以設定。關於 ifcfg-eth0 的設定則請參考第四章的說明。

不過，由於這兩支程式主要是搜尋設定檔 (ifcfg-ethx) 來進行啟動與關閉的， 所以在使用前請確定 ifcfg-ethx 是否真的存在於正確的目錄內，否則會啟動失敗喔！ 另外，如果以 ifconfig eth0 .... 來設定或者是修改了網路介面後， 那就無法再以 ifdown eth0 的方式來關閉了！因為 ifdown 會分析比對目前的網路參數與 ifcfg-eth0 是否相符，不符的話，就會放棄該次動作。因此，使用 ifconfig 修改完畢後，應該要以 ifconfig eth0 down 才能夠關閉該介面喔！

5.1.2 路由修改： route

我們在第二章網路基礎的時候談過關於路由的問題， 兩部主機之間一定要有路由才能夠互通 TCP/IP 的協定，否則就無法進行連線啊！一般來說，只要有網路介面， 該介面就會產生一個路由，所以我們安裝的主機有一個 eth0 的介面，看起來就會是這樣：

```
[root@www ~]# route [-nee]
[root@www ~]# route add [-net|-host] [網域或主機] netmask [mask] [gw|dev]
[root@www ~]# route del [-net|-host] [網域或主機] netmask [mask] [gw|dev]
觀察的參數：
  -n  ：不要使用通訊協定或主機名稱，直接使用 IP 或 port number；
  -ee ：使用更詳細的資訊來顯示
增加 (add) 與刪除 (del) 路由的相關參數：
  -net    ：表示後面接的路由為一個網域；
  -host   ：表示後面接的為連接到單部主機的路由；
  netmask ：與網域有關，可以設定 netmask 決定網域的大小；
  gw      ：gateway 的簡寫，後續接的是 IP 的數值喔，與 dev 不同；
  dev     ：如果只是要指定由那一塊網路卡連線出去，則使用這個設定，後面接 eth0 等

# 範例一：單純的觀察路由狀態
[root@www ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
0.0.0.0     192.168.1.254  0.0.0.0     UG   0    0     0 eth0

[root@www ~]# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     *               255.255.255.0   U     0      0        0 eth0
link-local      *               255.255.0.0     U     1002   0        0 eth0
default     192.168.1.254  0.0.0.0     UG   0    0     0 eth0
```

由上面的例子當中仔細觀察 route 與 route -n 的輸出結果，你可以發現有加 -n 參數的主要是顯示出 IP ，至於使用 route 而已的話，顯示的則是『主機名稱』喔！也就是說，在預設的情況下， route 會去找出該 IP 的主機名稱，如果找不到呢？ 就會顯示的鈍鈍的(有點小慢)，所以說，鳥哥通常都直接使用 route -n 啦！ 由上面看起來，我們也知道 default = 0.0.0.0/0.0.0.0 ， 而上面的資訊有哪些你必須要知道的呢？

- **Destination, Genmask**：這兩個玩意兒就是分別是 network 與 netmask 啦！所以這兩個咚咚就組合成為一個完整的網域囉！

- **Gateway**：該網域是通過哪個 gateway 連接出去的？如果顯示 0.0.0.0 表示該路由是直接由本機傳送，亦即可以透過區域網路的 MAC 直接傳訊；如果有顯示 IP 的話，表示該路由需要經過路由器 (通訊閘) 的幫忙才能夠傳送出去。

- **Flags**：總共有多個旗標，代表的意義如下：
  - U (route is up)：該路由是啟動的；
  - H (target is a host)：目標是一部主機 (IP) 而非網域；
  - G (use gateway)：需要透過外部的主機 (gateway) 來轉遞封包；
  - R (reinstate route for dynamic routing)：使用動態路由時，恢復路由資訊的旗標；
  - D (dynamically installed by daemon or redirect)：已經由服務或轉 port 功能設定為動態路由
  - M (modified from routing daemon or redirect)：路由已經被修改了；
  - ! (reject route)：這個路由將不會被接受(用來抵擋不安全的網域！)

- **Iface**：這個路由傳遞封包的介面。

此外，觀察一下上面的路由排列順序喔，依序是由小網域 (192.168.1.0/24 是 Class C)，逐漸到大網域 (169.254.0.0/16 Class B) 最後則是預設路由 (0.0.0.0/0.0.0.0)。 然後當我們要判斷某個網路封包應該如何傳送的時候，該封包會經由這個路由的過程來判斷喔！ 舉例來說，我上頭僅有三個路由，若我有一個傳往 192.168.1.20 的封包要傳遞，那首先會找 192.168.1.0/24 這個網域的路由，找到了！所以直接由 eth0 傳送出去；

如果是傳送到 Yahoo 的主機呢？ Yahoo 的主機 IP 是 119.160.246.241，我們通過判斷 1)不是 192.168.1.0/24， 2)不是 169.254.0.0/16 結果到達 3)0/0 時，OK！傳出去了，透過 eth0 將封包傳給 192.168.1.254 那部 gateway 主機啊！所以說，路由是有順序的。

因此當你重複設定多個同樣的路由時， 例如在你的主機上的兩張網路卡設定為相同網域的 IP 時，會出現什麼情況？會出現如下的情況：

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
```

也就是說，由於路由是依照順序來排列與傳送的， 所以不論封包是由那個介面 (eth0, eth1) 所接收，都會由上述的 eth0 傳送出去， 所以，在一部主機上面設定兩個相同網域的 IP 本身沒有什麼意義！有點多此一舉就是了。 除非是類似虛擬機器 (Xen, VMware 等軟體) 所架設的多主機時，才會有這個必要～

```
# 範例二：路由的增加與刪除
[root@www ~]# route del -net 169.254.0.0 netmask 255.255.0.0 dev eth0
```

```
# 上面這個動作可以刪除掉 169.254.0.0/16 這個網域！
# 請注意，在刪除的時候，需要將路由表上面出現的資訊都寫入
# 包括 netmask , dev 等等參數喔！注意注意

[root@www ~]# route add -net 192.168.100.0 \
> netmask 255.255.255.0 dev eth0
# 透過 route add 來增加一個路由！請注意，這個路由的設定必須要能夠與你的網路互通。
# 舉例來說，如果我下達底下的指令就會顯示錯誤：
# route add -net 192.168.200.0 netmask 255.255.255.0 gw 192.168.200.254
# 因為我的主機內僅有 192.168.1.11 這個 IP ，所以不能直接與 192.168.200.254
# 這個網段直接使用 MAC 互通！這樣說，可以理解嗎？

[root@www ~]# route add default gw 192.168.1.250
# 增加預設路由的方法！請注意，只要有一個預設路由就夠了喔！
# 同樣的，那個 192.168.1.250 的 IP 也需要能與你的 LAN 溝通才行！
# 在這個地方如果你隨便設定後，記得使用底下的指令重新設定你的網路
# /etc/init.d/network restart
```

如果是要進行路由的刪除與增加，那就得要參考上面的例子了，其實，使用 man route 裡面的資料就很豐富了！仔細查閱一下囉！ 你只要記得，當出現『SIOCADDRT: Network is unreachable』 這個錯誤時，肯定是由於 gw 後面接的 IP 無法直接與你的網域溝通 (Gateway 並不在你的網域內)， 所以，趕緊檢查一下是否輸入錯誤啊！

Tips:

一般來說，鳥哥如果接觸到一個新的環境內的主機，在不想要更動原系統的設定檔情況下，然後預計使用本書的網路環境設定時， 手動的處理就變成：『ifconfig eth0 192.168.1.100; route add default gw 192.168.1.254』這樣就搞定了！ 直接聯網與測試。等到完成測試後，再給她 /etc/init.d/network restart 恢復原系統的網路即可。

5.1.3 網路參數綜合指令： ip

ip 是個指令喔！並不是那個 TCP/IP 的 IP 啦！這個 ip 指令的功能可多了！基本上，他就是整合了 ifconfig 與 route 這兩個指令囉～不過， ip 可以達成的功能卻又多更多！真是個相當厲害的指令。如果你有興趣的話，請自行 vi /sbin/ifup ，就知道整個 ifup 就是利用 ip 這個指令來達成的。好了，如何使用呢？讓我們來瞧一瞧先！

```
[root@www ~]# ip [option] [動作] [指令]
選項與參數：
option ：設定的參數，主要有：
  -s ：顯示出該裝置的統計數據(statistics)，例如總接受封包數等；
動作：亦即是可以針對哪些網路參數進行動作，包括有：
  link ：關於裝置 (device) 的相關設定，包括 MTU, MAC 位址等等
  addr/address ：關於額外的 IP 協定，例如多 IP 的達成等等；
  route ：與路由有關的相關設定
```

由上面的語法我們可以知道， ip 除了可以設定一些基本的網路參數之外，還能夠進行額外的 IP 協定，包括多 IP 的達成，真是太完美了！底下我們就分三個部分 (link, addr, route) 來介紹這個 ip 指令吧！

- 關於裝置介面 (device) 的相關設定： ip link

ip link 可以設定與裝置 (device) 有關的相關參數，包括 MTU 以及該網路介面的 MAC 等等，當然也可以啟動 (up) 或關閉 (down) 某個網路介面啦！整個語法是這樣的：

```
[root@www ~]# ip [-s] link show    <== 單純的查閱該裝置相關的資訊
[root@www ~]# ip link set [device] [動作與參數]
選項與參數：
show：僅顯示出這個裝置的相關內容，如果加上 -s 會顯示更多統計數據；
set ：可以開始設定項目， device 指的是 eth0, eth1 等等介面代號；
動作與參數：包括有底下的這些動作：
  up|down  ：啟動 (up) 或關閉 (down) 某個介面，其他參數使用預設的乙太網路；
  address  ：如果這個裝置可以更改 MAC 的話，用這個參數修改！
  name     ：給予這個裝置一個特殊的名字；
  mtu      ：就是最大傳輸單元啊！

# 範例一：顯示出所有的介面資訊
[root@www ~]# ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:71:85:bd brd ff:ff:ff:ff:ff:ff
3: eth1: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN qlen 1000
    link/ether 08:00:27:2a:30:14 brd ff:ff:ff:ff:ff:ff
4: sit0: <NOARP> mtu 1480 qdisc noop state DOWN
    link/sit 0.0.0.0 brd 0.0.0.0

[root@www ~]# ip -s link show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:71:85:bd brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    314685     3354     0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    27200      199      0       0       0       0
```

使用 ip link show 可以顯示出整個裝置介面的硬體相關資訊，如上所示，包括網卡位址(MAC)、MTU等等， 比較有趣的應該是那個 sit0 的介面了，那個 sit0 的介面是用在 IPv4 及 IPv6 的封包轉換上的， 對於我們僅使用 IPv4 的網路是沒有作用的。 lo 及 sit0 都是主機內部所自行設定的。 而如果加上 -s 的參數後，則這個網路卡的相關統計資訊就會被列出來， 包括接收 (RX) 及傳送 (TX) 的封包數量等等，詳細的內容與 ifconfig 所輸出的結果相同的。

```
# 範例二：啟動、關閉與設定裝置的相關資訊
[root@www ~]# ip link set eth0 up
# 啟動 eth0 這個裝置介面；

[root@www ~]# ip link set eth0 down
# 阿就關閉啊！簡單的要命～

[root@www ~]# ip link set eth0 mtu 1000
# 更改 MTU 的值，達到 1000 bytes，單位就是 bytes 啊！
```

更新網路卡的 MTU 使用 ifconfig 也可以達成啊！沒啥了不起，不過，如果是要更改『網路卡代號、MAC 位址的資訊』的話，那可就得使用 ip 囉～不過，設定前可能得要先關閉該網路卡，否則會不成功。 如下所示：

```
# 範例三：修改網路卡代號、MAC 等參數
```

```
[root@www ~]# ip link set eth0 name vbird
SIOCSIFNAME: Device or resource busy
# 因為該裝置目前是啟動的，所以不能這樣做設定。你應該要這樣做：

[root@www ~]# ip link set eth0 down       <==關閉介面
[root@www ~]# ip link set eth0 name vbird  <==重新設定
[root@www ~]# ip link show            <==觀察一下
2: vbird: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:71:85:bd brd ff:ff:ff:ff:ff:ff
# 怕了吧！連網路卡代號都可以改變！不過，玩玩後記得改回來啊！
# 因為我們的 ifcfg-eth0 還是使用原本的裝置代號！避免有問題，要改回來

[root@www ~]# ip link set vbird name eth0  <==介面改回來

[root@www ~]# ip link set eth0 address aa:aa:aa:aa:aa:aa
[root@www ~]# ip link show eth0
# 如果你的網路卡支援硬體位址(MAC)可以更改的話，上面這個動作就可以更改
# 你的網路卡位址了！厲害吧！不過，還是那句老話，測試完之後請立刻改回來啊！
```

在這個裝置的硬體相關資訊設定上面，包括 MTU, MAC 以及傳輸的模式等等，都可以在這裡設定。 有趣的是那個 address 的項目，那個項目後面接的可是硬體位址 (MAC) 而不是 IP 喔！ 很容易搞錯啊！切記切記！更多的硬體參數可以使用 man ip 查閱一下與 ip link 有關的設定。

- **關於額外的 IP 相關設定： ip address**

如果說 ip link 是與 OSI 七層協定 的第二層資料連階層有關的話，那麼 ip address (ip addr) 就是與第三層網路層有關的參數啦！ 主要是在設定與 IP 有關的各項參數，包括 netmask, broadcast 等等。

```
[root@www ~]# ip address show      <==就是查閱 IP 參數啊！
[root@www ~]# ip address [add|del] [IP參數] [dev 裝置名] [相關參數]
選項與參數：
show    ：單純的顯示出介面的 IP 資訊啊；
add|del ：進行相關參數的增加 (add) 或刪除 (del) 設定，主要有：
  IP 參數：主要就是網域的設定，例如 192.168.100.100/24 之類的設定喔；
  dev   ：這個 IP 參數所要設定的介面，例如 eth0, eth1 等等；
  相關參數：主要有底下這些：
    broadcast：設定廣播位址，如果設定值是 + 表示『讓系統自動計算』
    label   ：亦即是這個裝置的別名，例如 eth0:0 就是了！
    scope   ：這個介面的領域，通常是這幾個大類：
          global ：允許來自所有來源的連線；
          site   ：僅支援 IPv6 ，僅允許本主機的連線；
          link   ：僅允許本裝置自我連線；
          host   ：僅允許本主機內部的連線；
          所以當然是使用 global 囉！預設也是 global 啦！

# 範例一：顯示出所有的介面之 IP 參數：
[root@www ~]# ip address show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:71:85:bd brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.100/24 brd 192.168.1.255 scope global eth0
    inet6 fe80::a00:27ff:fe71:85bd/64 scope link
       valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN qlen 1000
```

```
    link/ether 08:00:27:2a:30:14 brd ff:ff:ff:ff:ff:ff
4: sit0: <NOARP> mtu 1480 qdisc noop state DOWN
    link/sit 0.0.0.0 brd 0.0.0.0
```

看到上面那個特殊的字體嗎？沒錯！那就是 IP 參數啦！也是 ip address 最主要的功能。 底下我們進一步來新增虛擬的網路介面試看看：

```
# 範例二：新增一個介面，名稱假設為 eth0:vbird
[root@www ~]# ip address add 192.168.50.50/24 broadcast + \
> dev eth0 label eth0:vbird
[root@www ~]# ip address show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:71:85:bd brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.100/24 brd 192.168.1.255 scope global eth0
    inet 192.168.50.50/24 brd 192.168.50.255 scope global eth0:vbird
    inet6 fe80::a00:27ff:fe71:85bd/64 scope link
       valid_lft forever preferred_lft forever
# 看到上面的特殊字體了吧？多出了一行新的介面，且名稱是 eth0:vbird
# 至於那個 broadcast + 也可以寫成 broadcast 192.168.50.255 啦！

[root@www ~]# ifconfig
eth0:vbird Link encap:Ethernet  HWaddr 08:00:27:71:85:BD
          inet addr:192.168.50.50  Bcast:192.168.50.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
# 如果使用 ifconfig 就能夠看到這個怪東西了！可愛吧！ ^_^

# 範例三：將剛剛的介面刪除
[root@www ~]# ip address del 192.168.50.50/24 dev eth0
# 刪除就比較簡單啊！ ^_^
```

- **關於路由的相關設定： ip route**

這個項目當然就是路由的觀察與設定囉！事實上， ip route 的功能幾乎與 route 這個指令差不多，但是，他還可以進行額外的參數設計，例如 MTU 的規劃等等，相當的強悍啊！

```
[root@www ~]# ip route show    <==單純的顯示出路由的設定而已
[root@www ~]# ip route [add|del] [IP或網域] [via gateway] [dev 裝置]
選項與參數：
show ：單純的顯示出路由表，也可以使用 list ；
add|del ：增加 (add) 或刪除 (del) 路由的意思。
  IP或網域：可使用 192.168.50.0/24 之類的網域或者是單純的 IP ；
  via    ：從那個 gateway 出去，不一定需要；
  dev    ：由那個裝置連出去，這就需要了！
  mtu    ：可以額外的設定 MTU 的數值喔！

# 範例一：顯示出目前的路由資料
[root@www ~]# ip route show
192.168.1.0/24 dev eth0  proto kernel  scope link  src 192.168.1.100
169.254.0.0/16 dev eth0  scope link  metric 1002
default via 192.168.1.254 dev eth0
```

如上表所示，最簡單的功能就是顯示出目前的路由資訊，其實跟 route 這個指令相同啦！ 指示必須要注意幾個小東西：

- proto：此路由的路由協定，主要有 redirect, kernel, boot, static, ra 等， 其中 kernel 指的是直接由核心判斷自動設定。

- scope：路由的範圍，主要是 link ，亦即是與本裝置有關的直接連線。

再來看一下如何進行路由的增加與刪除吧！

```
# 範例二：增加路由，主要是本機直接可溝通的網域
[root@www ~]# ip route add 192.168.5.0/24 dev eth0
# 針對本機直接溝通的網域設定好路由，不需要透過外部的路由器
[root@www ~]# ip route show
192.168.5.0/24 dev eth0  scope link
....(以下省略)....

# 範例三：增加可以通往外部的路由，需透過 router 喔！
[root@www ~]# ip route add 192.168.10.0/24 via 192.168.5.100 dev eth0
[root@www ~]# ip route show
192.168.5.0/24 dev eth0  scope link
....(其他省略)....
192.168.10.0/24 via 192.168.5.100 dev eth0
# 仔細看喔，因為我有 192.168.5.0/24 的路由存在 (我的網卡直接聯繫)，
# 所以才可以將 192.168.10.0/24 的路由丟給 192.168.5.100
# 那部主機來幫忙傳遞喔！與之前提到的 route 指令是一樣的限制！

# 範例四：增加預設路由
[root@www ~]# ip route add default via 192.168.1.254 dev eth0
# 那個 192.168.1.254 就是我的預設路由器 (gateway) 的意思啊！^_^
# 真的記得，只要一個預設路由就 OK ！

# 範例五：刪除路由
[root@www ~]# ip route del 192.168.10.0/24
[root@www ~]# ip route del 192.168.5.0/24
```

事實上，這個 ip 的指令實在是太博大精深了！剛接觸 Linux 網路的朋友，可能會看到有點暈～ 不要緊啦！你先會使用 ifconfig, ifup , ifdown 與 route 即可， 等以後有經驗了之後，再繼續回來玩 ip 這個好玩的指令吧！ ^_^ 有興趣的話，也可以自行參考 ethtool 這個指令喔！ (man ethtool)。

5.1.4 無線網路： iwlist, iwconfig

這兩個指令你必須要有無線網卡才能夠進行喔！這兩個指令的用途是這樣的：

- iwlist：利用無線網卡進行無線 AP 的偵測與取得相關的資料；
- iwconfig：設定無線網卡的相關參數。

這兩個指令的應用我們在第四章裡面的無線網卡設定談了很多了， 所以這裡我們不再詳談，有興趣的朋友應該先使用 man iwlist 與 man iwconfig 瞭解一下語法， 然後再到前一章的無線網路小節查一查相關的用法，就瞭解了啦！ ^_^

5.1.5 手動使用 DHCP 自動取得 IP 參數： dhclient

如果你是使用 DHCP 協定在區域網路內取得 IP 的話，那麼是否一定要去編輯 ifcfg-eth0 內的 BOOTPROTO 呢？ 嘿嘿！有個更快速的作法，那就是利用 dhclient 這個指令～因為這個指令才是真正發送 dhcp 要求工作的程式啊！那要如何使用呢？很簡單！如果不考慮其他的參數，使用底下的方法即可：

```
[root@www ~]# dhclient eth0
```

夠簡單吧！這樣就可以立刻叫我們的網路卡以 dhcp 協定去嘗試取得 IP 喔！

5.2 網路偵錯與觀察指令

在網路的互助論壇中，最常聽到的一句話就是：『高手求救！我的 Linux 不能連上網路了！』我的天吶！不能上網路的原因多的很！而要完全搞懂也不是一件簡單的事情呢！ 不過，事實上我們可以自己使用測試軟體來追蹤可能的錯誤原因，而很多的網路偵測指令其實在 Linux 裡頭已經都預設存在了，只要你好好的學一學基本的偵測指令，那麼一些朋友在告訴你如何偵錯的時候， 你應該就立刻可以知道如何來搞定他囉！

其實我們在第四章談到的五個檢查步驟已經是相當詳細的網路偵錯流程了！ 只是還有些重要的偵測指令也得要來瞭解一下才好！

5.2.1 兩部主機兩點溝通： ping

這個 ping 是很重要的指令，ping 主要透過 ICMP 封包 來進行整個網路的狀況報告，當然啦，最重要的就是那個 ICMP type 0, 8 這兩個類型， 分別是要求回報與主動回報網路狀態是否存在的特性。要特別注意的是， ping 還是需要透過 IP 封包來傳送 ICMP 封包的， 而 IP 封包裡面有個相當重要的 TTL 屬性，這是很重要的一個路由特性， 詳細的 IP 與 ICMP 表頭資料請參考第二章網路基礎的詳細介紹。

```
[root@www ~]# ping [選項與參數] IP
選項與參數：
-c 數值：後面接的是執行 ping 的次數，例如 -c 5 ；
-n    ：在輸出資料時不進行 IP 與主機名稱的反查，直接使用 IP 輸出(速度較快)；
-s 數值：發送出去的 ICMP 封包大小，預設為 56bytes，不過你可以放大此一數值；
-t 數值：TTL 的數值，預設是 255，每經過一個節點就會少一；
-W 數值：等待回應對方主機的秒數。
-M [do|dont] ：主要在偵測網路的 MTU 數值大小，兩個常見的項目是：
  do  ：代表傳送一個 DF (Don't Fragment) 旗標，讓封包不能重新拆包與打包；
  dont：代表不要傳送 DF 旗標，表示封包可以在其他主機上拆包與打包

# 範例一：偵測一下 168.95.1.1 這部 DNS 主機是否存在？
[root@www ~]# ping -c 3 168.95.1.1
PING 168.95.1.1 (168.95.1.1) 56(84) bytes of data.
64 bytes from 168.95.1.1: icmp_seq=1 ttl=245 time=15.4 ms
64 bytes from 168.95.1.1: icmp_seq=2 ttl=245 time=10.0 ms
64 bytes from 168.95.1.1: icmp_seq=3 ttl=245 time=10.2 ms

--- 168.95.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2047ms
rtt min/avg/max/mdev = 10.056/11.910/15.453/2.506 ms
```

ping 最簡單的功能就是傳送 ICMP 封包去要求對方主機回應是否存在於網路環境中，上面的回應訊息當中，幾個重要的項目是這樣的：

- **64 bytes**：表示這次傳送的 ICMP 封包大小為 64 bytes 這麼大，這是預設值， 在某些特殊場合中，例如要搜索整個網路內最大的 MTU 時，可以使用 -s 2000 之類的數值來取代；

- **icmp_seq=1**：ICMP 所偵測進行的次數，第一次編號為 1 ；
- **ttl=243**：TTL 與 IP 封包內的 TTL 是相同的，每經過一個帶有 MAC 的節點 (node) 時，例如 router, bridge 時， TTL 就會減少一，預設的 TTL 為 255 ， 你可以透過 -t 150 之類的方法來重新設定預設 TTL 數值；
- **time=15.4 ms**：回應時間，單位有 ms(0.001秒)及 us(0.000001秒)， 一般來說，越小的回應時間，表示兩部主機之間的網路連線越良好！

如果你忘記加上 -c 3 這樣的規定偵測次數，那就得要使用 [ctrl]-c 將他結束掉了！

例題：

寫一支腳本程式 ping.sh ，透過這支腳本程式，你可以用 ping 偵測整個網域的主機是否有回應。此外，每部主機的偵測僅等待一秒鐘，也僅偵測一次。

答：

由於僅偵測一次且等待一秒，因此 ping 的選項為： -W1 -c1 ，而位於本機所在的區網為 192.168.1.0/24 ，所以可以這樣寫 (vim /root/bin/ping.sh)：

```
#!/bin/bash
for siteip in $(seq 1 254)
do
        site="192.168.1.${siteip}"
        ping -c1 -W1 ${site} &> /dev/null
        if [ "$?" == "0" ]; then
                echo "$site is UP"
        else
                echo "$site is DOWN"
        fi
done
```

特別注意一下，如果你的主機與待偵測主機並不在同一個網域內， 那麼 TTL 預設使用 255 ，如果是同一個網域內，那麼 TTL 預設則使用 64 喔！

- 用 **ping** 追蹤路徑中的最大 **MTU** 數值

我們由第二章的網路基礎裡面談到加大訊框 (frame) 時， 對於網路效能是有幫助的，因為封包打包的次數會減少，加上如果整個傳輸的媒體都能夠接受這個 frame 而不需要重新進行封包的拆解與重組的話，那麼效能當然會更好，那個修改 frame 大小的參數就是 MTU 啦！

好了，現在我們知道網路卡的 MTU 修改可以透過 ifconfig 或者是 ip 等指令來達成，那麼追蹤整個網路傳輸的最大 MTU 時，又該如何查詢？呵呵！最簡單的方法當然是透過 ping 傳送一個大封包， 並且不許中繼的路由器或 switch 將該封包重組，那就能夠處理啦！沒錯！可以這樣的：

```
# 範例二：找出最大的 MTU 數值
[root@www ~]# ping -c 2 -s 1000 -M do 192.168.1.254
```

```
PING 192.168.1.254 (192.168.1.254) 1000(1028) bytes of data.
1008 bytes from 192.168.1.254: icmp_seq=1 ttl=64 time=0.311 ms
# 如果有回應，那就是可以接受這個封包，如果無回應，那就表示這個 MTU 太大了。

[root@www ~]# ping -c 2 -s 8000 -M do 192.168.1.254
PING 192.168.1.254 (192.168.1.254) 8000(8028) bytes of data.
From 192.168.1.100 icmp_seq=1 Frag needed and DF set (mtu = 1500)
# 這個錯誤訊息是說，本地端的 MTU 才到 1500 而已，你要偵測 8000 的 MTU
# 根本就是無法達成的！那要如何是好？用前一小節介紹的 ip link 來進行 MTU 設定吧！
```

不過，你需要知道的是，由於 IP 封包表頭 (不含 options) 就已經佔用了 20 bytes ，再加上 ICMP 的表頭有 8 bytes ，所以當然你在使用 -s size 的時候，那個封包的大小就得要先扣除 (20+8=28) 的大小了。 因此如果要使用 MTU 為 1500 時，就得要下達『 ping -s 1472 -M do xx.yy.zz.ip 』才行啊！

另外，由於本地端的網路卡 MTU 也會影響到偵測，所以如果想要偵測整個傳輸媒體的 MTU 數值，那麼每個可以調整的主機就得要先使用 ifcofig 或 ip 先將 MTU 調大，然後再去進行偵測， 否則就會出現像上面提供的案例一樣，可能會出現錯誤訊息的！

不過這個 MTU 不要隨便調整啊！除非真的有問題。通常調整 MTU 的時間是在這個時候：

- 因為全部的主機群都是在內部的區網，例如叢集架構 (cluster) 的環境下， 由於內部的網路節點都是我們可以控制的，因此可以透過修改 MTU 來增進網路效能；
- 因為作業系統預設的 MTU 與你的網域不符，導致某些網站可以順利連線，某些網站則無法連線。 以 Windows 作業系統作為連線分享的主機時，在 Client 端挺容易發生這個問題；

如果是要連上 Internet 的主機，注意不要隨便調整 MTU ，因為我們無法知道 Internet 上面的每部機器能夠支援的 MTU 到多大，因為.....不是我們能夠管的到的嘛 ^_^！ 另外，其實每種連線方式都有不同的 MTU 值，常見的各種介面的 MTU 值分別為：

網路介面	MTU
Ethernet	1500
PPPoE	1492
Dial-up(Modem)	576

5.2.2 兩主機間各節點分析： traceroute

我們前面談到的指令大多數都是針對主機的網路參數設定所需要的，而 ping 是兩部主機之間的回聲與否判斷， 那麼有沒有指令可以追蹤兩部主機之間通過的各個節點 (node) 通訊狀況的好壞呢？舉例來說，如果我們連線到 yahoo 的速度比平常慢，你覺得是 (1)自己的網路環境有問題？ (2)還是外部的 Internet 有問題？如果是 (1) 的話，我們當然需要檢查自己的網路環境啊，看看是否又有誰中毒了？但如果是 Internet 的問題呢？那只有『等等等』啊！ 判斷是 (1) 還是 (2) 就得要使用 traceroute 這個指令啦！

```
[root@www ~]# traceroute [選項與參數] IP
選項與參數：
-n ：可以不必進行主機的名稱解析，單純用 IP ，速度較快！
-U ：使用 UDP 的 port 33434 來進行偵測，這是預設的偵測協定；
```

```
-I ：使用 ICMP 的方式來進行偵測；
-T ：使用 TCP 來進行偵測，一般使用 port 80 測試
-w ：若對方主機在幾秒鐘內沒有回聲就宣告不治...預設是 5 秒
-p 埠號：若不想使用 UDP 與 TCP 的預設埠號來偵測，可在此改變埠號。
-i 裝置：用在比較複雜的環境，如果你的網路介面很多很複雜時，才會用到這個參數；
      舉例來說，你有兩條 ADSL 可以連接到外部，那你的主機會有兩個 ppp，
      你可以使用 -i 來選擇是 ppp0 還是 ppp1 啦！
-g 路由：與 -i 的參數相仿，只是 -g 後面接的是 gateway 的 IP 就是了。

# 範例一：偵測本機到 yahoo 去的各節點連線狀態
[root@www ~]# traceroute -n tw.yahoo.com
traceroute to tw.yahoo.com (119.160.246.241), 30 hops max, 40 byte packets
 1  192.168.1.254  0.279 ms  0.156 ms  0.169 ms
 2  172.20.168.254  0.430 ms  0.513 ms  0.409 ms
 3  10.40.1.1  0.996 ms  0.890 ms  1.042 ms
 4  203.72.191.85  0.942 ms  0.969 ms  0.951 ms
 5  211.20.206.58  1.360 ms  1.379 ms  1.355 ms
 6  203.75.72.90  1.123 ms  0.988 ms  1.086 ms
 7  220.128.24.22  11.238 ms  11.179 ms  11.128 ms
 8  220.128.1.82  12.456 ms  12.327 ms  12.221 ms
 9  220.128.3.149  8.062 ms  8.058 ms  7.990 ms
10  * * *
11  119.160.240.1  10.688 ms  10.590 ms 119.160.240.3  10.047 ms
12  * * * <==可能有防火牆裝置等情況發生所致
```

這個 traceroute 挺有意思的，這個指令會針對欲連接的目的地之所有 node 進行 UDP 的逾時等待， 例如上面的例子當中，由鳥哥的主機連接到 Yahoo 時，他會經過 12 個節點以上，traceroute 會主動的對這 12 個節點做 UDP 的回聲等待，並偵測回覆的時間，每節點偵測三次，最終回傳像上頭顯示的結果。 你可以發現每個節點其實回覆的時間大約在 50 ms 以內，算是還可以的 Internet 環境了。

比較特殊的算是第 10/12 個，會回傳星號的，代表該 node 可能設有某些防護措施，讓我們發送的封包資訊被丟棄所致。 因為我們是直接透過路由器轉遞封包，並沒有進入路由器去取得路由器的使用資源，所以某些路由器僅支援封包轉遞， 並不會接受來自用戶端的各項偵測啦！此時就會出現上述的問題。因為 traceroute 預設使用 UDP 封包，如果你想嘗試使用其他封包， 那麼 -I 或 -T 可以試看看囉！

由於目前 UDP/ICMP 的攻擊層出不窮，因此很多路由器可能就此取消這兩個封包的回應功能。所以我們可以使用 TCP 來偵測呦！ 例如使用同樣的方法，透過等待時間 1 秒，以及 TCP 80 埠口的情況下，可以這樣做：

```
[root@www ~]# traceroute -w 1 -n -T tw.yahoo.com
```

5.2.3 察看本機的網路連線與後門： netstat

如果你覺得你的某個網路服務明明就啟動了，但是就是無法造成連線的話，那麼應該怎麼辦？ 首先你應該要查詢一下自己的網路介面所監聽的埠口 (port) 來看看是否真的有啟動，因為有時候螢幕上面顯示的 [OK] 並不一定是 OK 啊！ ^_^

```
[root@www ~]# netstat -[rn]       <==與路由有關的參數
[root@www ~]# netstat -[antulpc]   <==與網路介面有關的參數
選項與參數：
與路由 (route) 有關的參數說明：
-r ：列出路由表(route table)，功能如同 route 這個指令；
```

```
-n ：不使用主機名稱與服務名稱，使用 IP 與 port number ，如同 route -n
與網路介面有關的參數：
-a ：列出所有的連線狀態，包括 tcp/udp/unix socket 等；
-t ：僅列出 TCP 封包的連線；
-u ：僅列出 UDP 封包的連線；
-l ：僅列出有在 Listen (監聽) 的服務之網路狀態；
-p ：列出 PID 與 Program 的檔名；
-c ：可以設定幾秒鐘後自動更新一次，例如 -c 5 每五秒更新一次網路狀態的顯示；

# 範例一：列出目前的路由表狀態，且以 IP 及 port number 顯示：
[root@www ~]# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.1.0     0.0.0.0         255.255.255.0   U         0 0          0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U         0 0          0 eth0
0.0.0.0         192.168.1.254   0.0.0.0         UG        0 0          0 eth0
# 其實這個參數就跟 route -n 一模一樣，對吧！這不是 netstat 的主要功能啦！

# 範例二：列出目前的所有網路連線狀態，使用 IP 與 port number
[root@www ~]# netstat -an
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address     Foreign Address     State
....(中間省略)....
tcp        0      0 127.0.0.1:25      0.0.0.0:*           LISTEN
tcp     0   52 192.168.1.100:22  192.168.1.101:1937  ESTABLISHED
tcp        0      0 :::22               :::*                LISTEN
....(中間省略)....
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags    Type    State      I-Node Path
unix  2      [ ACC ]  STREAM  LISTENING  11075  @/var/run/hald/dbus-uukdg1qMPh
unix  2      [ ACC ]  STREAM  LISTENING  10952  /var/run/dbus/system_bus_socket
unix  2      [ ACC ]  STREAM  LISTENING  11032  /var/run/acpid.socket
....(底下省略)....
```

netstat 的輸出主要分為兩大部分，分別是 TCP/IP 的網路介面部分，以及傳統的 Unix socket 部分。 還記得我們在基礎篇裡面曾經談到檔案的類型嗎？那個 socket 與 FIFO 檔案還記得吧？ 那就是在 Unix 介面用來做為程式資料交流的介面了，也就是上頭表格內看到的 Active Unix domain sockets 的內容囉～

通常鳥哥都是建議加上『 -n 』這個參數的，因為可以避過主機名稱與服務名稱的反查，直接以 IP 及埠口號碼 (port number) 來顯示，顯示的速度上會快很多！至於在輸出的訊息當中， 我們先來談一談關於網路連線狀態的輸出部分，他主要是分為底下幾個大項：

- **Proto**：該連線的封包協定，主要為 TCP/UDP 等封包；
- **Recv-Q**：非由使用者程式連接所複製而來的總 bytes 數；
- **Send-Q**：由遠端主機所傳送而來，但不具有 ACK 標誌的總 bytes 數， 意指主動連線 SYN 或其他標誌的封包所佔的 bytes 數；
- **Local Address**：本地端的位址，可以是 IP (-n 參數存在時)， 也可以是完整的主機名稱。使用的格是就是『 IP:port 』只是 IP 的格式有 IPv4 及 IPv6 的差異。 如上所示，在 port 22 的介面中，使用的 :::22 就是針對 IPv6 的顯示，事實上他就相同於 0.0.0.0:22 的意思。 至於 port 25 僅針對 lo 介面開放，意指 Internet 基本上是無法連接到我本機的 25 埠口啦！
- **Foreign Address**：遠端的主機 IP 與 port number
- **stat**：狀態列，主要的狀態含有：
  - ESTABLISED：已建立連線的狀態；
  - SYN_SENT：發出主動連線 (SYN 標誌) 的連線封包；
  - SYN_RECV：接收到一個要求連線的主動連線封包；
  - FIN_WAIT1：該插槽服務(socket)已中斷，該連線正在斷線當中；

- FIN_WAIT2：該連線已掛斷，但正在等待對方主機回應斷線確認的封包；
- TIME_WAIT：該連線已掛斷，但 socket 還在網路上等待結束；
- LISTEN：通常用在服務的監聽 port ！可使用『 -l 』參數查閱。

基本上，我們常常談到的 netstat 的功能，就是在觀察網路的連線狀態了，而網路連線狀態中， 又以觀察『我目前開了多少的 port 在等待用戶端的連線』以及 『目前我的網路連線狀態中，有多少連線已建立或產生問題』最常見。 那你如何瞭解與觀察呢？通常鳥哥是這樣處理的：

```
# 範例三：秀出目前已經啟動的網路服務
[root@www ~]# netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address  Foreign Address   State   PID/Program name
tcp        0      0 0.0.0.0:34796  0.0.0.0:*         LISTEN 987/rpc.statd
tcp        0      0 0.0.0.0:111    0.0.0.0:*         LISTEN 969/rpcbind
tcp        0      0 127.0.0.1:25   0.0.0.0:*         LISTEN 1231/master
tcp        0      0 :::22          :::*              LISTEN 1155/sshd
udp        0      0 0.0.0.0:111    0.0.0.0:*                969/rpcbind
....(底下省略)....
# 上面最重要的其實是那個 -l 的參數，因為可以僅列出有在 Listen 的 port
```

你可以發現很多的網路服務其實僅針對本機的 lo 開放而已，網際網路是連接不到該埠口與服務的。 而由上述的資料我們也可以看到，啟動 port 111 的，其實就是 rpcbind 那隻程式，那如果想要關閉這個埠口， 你可以使用 kill 刪除 PID 969，也可以使用 killall 刪除 rpcbind 這個程序即可。如此一來， 很輕鬆的你就能知道哪個程式啟動了哪些埠口囉！

```
# 範例四：觀察本機上頭所有的網路連線狀態
[root@www ~]# netstat -atunp
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address     Foreign Address     State       PID/Program
tcp        0      0 0.0.0.0:111       0.0.0.0:*           LISTEN      969/rpcbind
tcp        0      0 0.0.0.0:22        0.0.0.0:*           LISTEN      1155/sshd
tcp        0      0 127.0.0.1:25      0.0.0.0:*           LISTEN      1231/master
tcp        0     52 192.168.1.100:22  192.168.1.101:1937  ESTABLISHED 4716/0
....(底下省略)....
```

看到上頭的特殊字體吧？那代表目前已經建立連線的一條網路連線，他是由遠端主機 192.168.1.101 啟動一個大於 1024 的埠口向本地端主機 192.168.1.100 的 port 22 進行的一條連線， 你必須要想起來的是：『Client 端是隨機取一個大於 1024 以上的 port 進行連線』，此外『只有 root 可以啟動小於 1024 以下的 port 』，那就看的懂上頭那條連線囉！如果這條連線你想要砍掉他的話， 看到最右邊的 4716 了沒？ kill 會用吧！ ^_^

至於傳統的 Unix socket 的資料，記得使用 man netstat 查閱一下吧！ 這個 Unix socket 通常是用在一些僅在本機上運作的程式所開啟的插槽介面檔， 例如 X Window 不都是在本機上運作而已嗎？那何必啟動網路的 port 呢？當然可以使用 Unix socket 囉，另外，例如 Postfix 這一類的網路伺服器，由於很多動作都是在本機上頭來完成的， 所以以會佔用很多的 Unix socket 喔！

例題：

請說明服務名稱與 port number 的對應在 Linux 當中，是用那個檔案來設定對應的？

答：

/etc/services

5.2.4 偵測主機名稱與 IP 對應： host, nslookup

關於主機名稱與 IP 的對應中，我們主要介紹的是 DNS 用戶端功能的 dig 這個指令。不過除了這個指令之外， 其實還有兩個更簡單的指令，那就是 host 與 nslookup 啦！底下讓我們來聊聊這兩個指令吧！

host

這個指令可以用來查出某個主機名稱的 IP 喔！舉例來說，我們想要知道 tw.yahoo.com 的 IP 時，可以這樣做：

```
[root@www ~]# host [-a] hostname [server]
選項與參數：
-a ：列出該主機詳細的各項主機名稱設定資料
[server] ：可以使用非為 /etc/resolv.conf 的 DNS 伺服器 IP 來查詢。

# 範例一：列出 tw.yahoo.com 的 IP
[root@www ~]# host tw.yahoo.com
tw.yahoo.com is an alias for tw-cidr.fyap.b.yahoo.com.
tw-cidr.fyap.b.yahoo.com is an alias for tw-tpe-fo.fyap.b.yahoo.com.
tw-tpe-fo.fyap.b.yahoo.com has address 119.160.246.241
```

瞧！IP 是 119.160.246.241 啊！很簡單就可以查詢到 IP 了！那麼這個 IP 是向誰查詢的呢？其實就是寫在 /etc/resolv.conf 那個檔案內的 DNS 伺服器 IP 啦！如果不想要使用該檔案內的主機來查詢，也可以這樣做：

```
[root@www ~]# host tw.yahoo.com 168.95.1.1
Using domain server:
Name: 168.95.1.1
Address: 168.95.1.1#53
Aliases:

tw.yahoo.com is an alias for tw-cidr.fyap.b.yahoo.com.
tw-cidr.fyap.b.yahoo.com is an alias for tw-tpe-fo.fyap.b.yahoo.com.
tw-tpe-fo.fyap.b.yahoo.com has address 119.160.246.241
```

會告訴我們所使用來查詢的主機是哪一部吶！這樣就夠清楚了吧！不過，再怎麼清楚也比不過 dig 這個指令的，所以這個指令僅是參考參考啦！

nslookup

這玩意兒的用途與 host 基本上是一樣的，就是用來作為 IP 與主機名稱對應的檢查， 同樣是使用 /etc/resolv.conf 這個檔案來作為 DNS 伺服器的來源選擇。

```
[root@www ~]# nslookup [-query=[type]] [hostname|IP]
選項與參數：
-query=type：查詢的類型，除了傳統的 IP 與主機名稱對應外，DNS 還有很多資訊，
        所以我們可以查詢很多不同的資訊，包括 mx, cname 等等，
        例如： -query=mx 的查詢方法！
```

```
# 範例一：找出 www.google.com 的 IP
[root@www ~]# nslookup www.google.com
Server:        168.95.1.1
Address:       168.95.1.1#53

Non-authoritative answer:
www.google.com  canonical name = www.l.google.com.
Name:   www.l.google.com
Address: 74.125.71.106
....(底下省略)....

# 範例二：找出 168.95.1.1 的主機名稱
[root@www ~]# nslookup 168.95.1.1
Server:         168.95.1.1
Address:        168.95.1.1#53

1.1.95.168.in-addr.arpa name = dns.hinet.net.
```

如何，看起來與 host 差不多吧！不過，這個 nslookup 還可以由 IP 找出主機名稱喔！ 例如那個範例二，他的主機名稱是： dns.hinet.net 哩！目前大家都建議使用 dig 這個指令來取代 nslookup ，我們會在第十九章 DNS 伺服器那時再來好好談一談吧！

5.3 遠端連線指令與即時通訊軟體

啥是遠端連線呢？其實就是在不同的電腦之間進行登入的情況啦！我們可以透過 telnet, ssh 或者是 ftp 等協定來進行遠端主機的登入。底下我們就分別來介紹一下這些基本的指令吧！這裡僅是談到用戶端功能喔，相關的伺服器我們則會在後續進行說明的。

5.3.1 終端機與 BBS 連線： telnet

telnet 是早期我們在個人電腦上面要連結到伺服器工作時，最重要的一個軟體了！他不但可以直接連接到伺服器上頭， 還可以用來連結 BBS 呢！非常棒！不過， telnet 本身的資料在傳送的時候是使用明碼 (原始的資料，沒有加密) ， 所以資料在 Internet 上面跑的時候，會比較危險一點 (就怕被別人監聽啊)。更詳細的資料我們會在第十一章遠端連線伺服器內做介紹的。

```
[root@www ~]# telnet [host|IP [port]]

# 範例一：連結到台灣相當熱門的 PTT BBS 站 ptt.cc
[root@www ~]# yum install telnet   <==預設沒有安裝這軟體
[root@www ~]# telnet ptt.cc
    歡迎來到 批踢踢實業坊 目前有【100118】名使用者與您一同對抗炎炎夏日。

請輸入代號，或以 guest 參觀，或以 new 註冊:
[高手召集令] 台灣駭客年會 暑假與你駭翻南港 http://reg.hitcon.org/hit2011
要學電腦，首選台灣大學資訊訓練班!  http://tinyurl.com/3z42apw
```

如上所示，我們可以透過 telnet 輕易的連結到 BBS 上面，而如果你的主機有開啟 telnet 伺服器服務的話，同樣的利用『 telnet IP 』並且輸入帳號與密碼之後，就能夠登入主機了。 另外，在 Linux 上的 telnet 軟體還提供了 Kerberos 的認證方式，有興趣的話請自行參閱 man telnet 的說明。

除了連結到伺服器以及連結到 BBS 站之外， telnet 還可以用來連結到某個 port (服務) 上頭吶！ 舉例來說，我們可以用 telnet 連接到 port 110 ，看看這個 port 是否有正確的啟動呢？

```
# 範例二：偵測本機端的 110 這個 port 是否正確啟動？
[root@www ~]# telnet localhost 110
Trying 127.0.0.1...
telnet: connect to address 127.0.0.1: Connection refused
# 如果出現這樣的訊息，代表這個 port 沒有啟動或者是這個連線有問題，
# 因為你看到那個 refused 嘛！

[root@www ~]# telnet localhost 25
Trying ::1...
Connected to localhost.
Escape character is '^]'.
220 www.centos.vbird ESMTP Postfix
ehlo localhost
250-www.centos.vbird
250-PIPELINING
250-SIZE 10240000
....(中間省略)....
250 DSN
quit
221 2.0.0 Bye
Connection closed by foreign host.
```

瞧！根據輸出的結果，我們就能夠知道這個通訊協定 (port number 提供的通訊協定功能) 是否有成功的啟動吶！ 而在每個 port 所監聽的服務都有其特殊的指令，例如上述的 port 25 就是在本機介面所提供的電子郵件服務， 那個服務所支援的指令就如同上面使用的資料一樣，但是其他的 port 就不見得支援這個『 ehlo 』的命令， 因為不同的 port 有不同的程式嘛！所以當然支援的命令就不同囉！

5.3.2 FTP 連線軟體： ftp, lftp

現在的人們由於有高容量的 email 可以用，因此傳送檔案可以很輕鬆的透過 email 。不過 email 還是有單封信件容量限制， 如果想要一口氣傳送個幾百 MB 的檔案，恐怕還是得要透過 FTP 這個通訊協定才行啊！文字介面的 FTP 軟體主要有 ftp, lftp 兩個，圖形介面的呢？在 CentOS 上面預設有 gftp 這個好用的東東。在這裡我們僅介紹文字介面的兩個指令而已。

---

ftp

ftp 這個指令很簡單，用在處理 FTP 伺服器的下載資料啦。由於鳥哥所在的位置在崑山科大，因此這裡使用崑山科大的 FTP 伺服器為例：

```
[root@www ~]# ftp [host|IP] [port]

# 範例一：連線到崑山科大去看看
[root@www ~]# yum install ftp
[root@www ~]# ftp ftp.ksu.edu.tw
Connected to ftp.ksu.edu.tw (120.114.150.21).
220---------- Welcome to Pure-FTPd [privsep] ----------
220-You are user number 1 of 50 allowed.
220-Local time is now 16:25. Server port: 21.
220-Only anonymous FTP is allowed here    <==訊息要看啊！這個 FTP 僅支援匿名
220-IPv6 connections are also welcome on this server.
220 You will be disconnected after 5 minutes of inactivity.
```

```
Name (ftp.ksu.edu.tw:root): anonymous   <==鳥哥這裡用匿名登入！
230 Anonymous user logged in              <==嗯！確實是匿名登入了！
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>                    <==最終登入的結果看起來是這樣！
ftp> help         <==提供需要的指令說明，可以常參考！
ftp> dir                <==顯示遠端伺服器的目錄內容 (檔名列表)
ftp> cd /pub             <==變換目錄到 /pub 當中
ftp> get filename     <==下載單一檔案，檔名為 filename
ftp> mget filename*  <==下載多個檔案，可使用萬用字元 *
ftp> put filename     <==上傳 filename 這個檔案到伺服器上
ftp> delete file      <==刪除主機上的 file 這個檔案
ftp> mkdir dir         <==建立 dir 這個目錄
ftp> lcd /home         <==切換『本地端主機』的工作目錄
ftp> passive           <==啟動或關閉 passive 模式
ftp> binary            <==資料傳輸模式設定為 binary 格式
ftp> bye                 <==結束 ftp 軟體的使用
```

FTP 其實算是一個很麻煩的協定，因為他使用兩個 port 分別進行命令與資料的交流，詳細的資料我們會在第二十一章的 FTP 伺服器內詳談，這裡我們先單純的介紹一下如何使用 ftp 這個軟體。首先我們當然是需要登入囉， 所以在上頭的表格當中我們當然需要填入帳號與密碼了。不過由於崑山科大僅提供匿名登入，而匿名登入者的帳號就是『 anonymous 』所以直接填寫那個帳號即可。如果是私人的 FTP 時，才需要提供一組完整的帳號與密碼啦！

登入 FTP 主機後，就能夠使用 ftp 軟體的功能進行上傳與下載的動作，幾個常用的 ftp 內指令如上表，不過，鳥哥建議你可以連到大學的 FTP 網站後，使用 help (或問號 ?) 來參考可用的指令，然後嘗試下載以測試使用一下這個指令吧！這樣以後沒有瀏覽器的時候，你也可以到 ftp 下載了呢！不錯吧！另外你要注意的是，離開 ftp 軟體時，得要輸入『 bye 』喔！不是『 exit 』啦！

如果由於某些理由，讓你的 FTP 主機的 port 開在非正規的埠口時，那你就可以利用底下的方式來連接到該部主機喔！

```
[root@www ~]# ftp hostname 318
# 假設對方主機的 ftp 服務開啟在 318 這個 port 啊！
```

- ## lftp (自動化腳本)

單純使用 ftp 總是覺得很麻煩，有沒有更快速的 ftp 用戶軟體呢？讓我們可以使用類似網址列的方式來登入 FTP 伺服器啊？有的，那就是 lftp 的功能了！ lftp 預設使用匿名登入 FTP 伺服器，可以使用類似網址列的方式取得資料， 使用上比單純的 ftp 要好用些。此外，由於可在指令列輸入帳號/密碼，可以輔助進行程式腳本的設計喔！

```
[root@www ~]# lftp [-p port] [-u user[,pass]] [host|IP]
[root@www ~]# lftp -f filename
[root@www ~]# lftp -c "commands"
選項與參數：
-p ：後面可以直接接上遠端 FTP 主機提供的 port
-u ：後面則是接上帳號與密碼，就能夠連接上遠端主機了
    如果沒有加帳號密碼， lftp 預設會使用 anonymous 嘗試匿名登入
-f ：可以將指令寫入腳本中，這樣可以幫助進行 shell script 的自動處理喔！
```

```
-c ：後面直接加上所需要的指令。

# 範例一：利用 lftp 登入崑山科大的 FTP 伺服器
[root@www ~]# yum install lftp
[root@www ~]# lftp ftp.ksu.edu.tw
lftp ftp.ksu.edu.tw:~>
# 瞧！一下子就登入了！很快樂吧！ ^_^！你同樣可使用 help 去查閱相關內部指令
```

至於登入 FTP 主機後，一樣可以使用『help』來顯示出可以執行的指令，與 ftp 很類似啦！不過多了書籤的功能，而且也非常的類似 bash 吶！很不錯呦！除了這個好用的文字介面的 FTP 軟體之外，事實上還有很多圖形介面的好用軟體呢！ 最常見的就是 gftp 了，非常的容易上手喔！ CentOS 本身就有提供 gftp 了，你可以拿出原版的光碟來安裝，然後進入 X Window 後， 啟動一個 shell ，輸入『 gftp 』就能夠發現他的好用啦！

如果你想要定時的去捉下崑山科大 FTP 網站下的 /pub/CentOS/RPM-GPG* 的檔案時，那麼那個腳本應該要怎麼寫呢？ 我們嘗試來寫寫看吧！

```
# 使用檔案配合 lftp 去處理時：
[root@www ~]# mkdir lftp; cd lftp
[root@www lftp]# vim lftp.ksu.sh
open ftp.ksu.edu.tw
cd /pub/CentOS/
mget -c -d RPM-GPG*
bye
[root@www lftp]# lftp -f lftp.ksu.sh
[root@www lftp]# ls
lftp.ksu.sh       RPM-GPG-KEY-CentOS-3 RPM-GPG-KEY-CentOS-4 RPM-GPG-KEY-CentOS-6
RPM-GPG-KEY-beta RPM-GPG-KEY-centos4  RPM-GPG-KEY-CentOS-5

# 直接將要處理的動作加入 lftp 指令中
[root@www lftp]# vim lftp.ksu.sh
lftp -c "open ftp.ksu.edu.tw
cd /pub/CentOS/
mget -c -d RPM-GPG*
bye"
[root@www lftp]# sh lftp.ksu.sh
```

若為非匿名登入時，則可以使用『 open -u username,password hostname 』修改 lftp.ksu.sh 的第一行！ 如果再將這個腳本寫入 crontab 當中，你就可以定時的以 FTP 進行上傳/下載的功能囉！這就是文字指令的好處！

5.3.3 圖形介面的即時通訊軟體： pidgin (gaim 的延伸)

現在應該大家都知道什麼是 MSN, 雅虎即時通以及其他的通訊軟體吧？那麼要連上這些伺服器時，該怎麼處理哪？很簡單，在 X Window 底下使用 pidgin 就好了！簡直簡單到不行～請先進入 X Window 系統，然後經過『應用程式』--> 『網際網路』-->『Pidgin 網路即時通』啟動他即可 (請注意你必須已經安裝了 pidgin 了，可用 yum install pidgin 處理)。

不過，傷腦筋的是，我們所安裝的 basic server 類型的 CentOS 6.x 主要做為伺服器之用，所以連圖形介面也沒有給我們。 所以，鳥哥又用另外一部主機安裝成 Desktop 的模式，利用該部主機來測試 pidgin 這玩意兒的！因此， 底下的練習你也可以先略過，等到你安裝另一部 Desktop linux 時再來玩玩！

圖 5.3-1、pidgin 的歡迎畫面

在上圖中按下『新增』，然後你會看到如下的畫面：

圖 5.3-2、pidgin 支援的即時通訊資料

很神奇的是， pidgin 支援的通訊有夠多的！我們使用 MSN 來作個解釋好了：

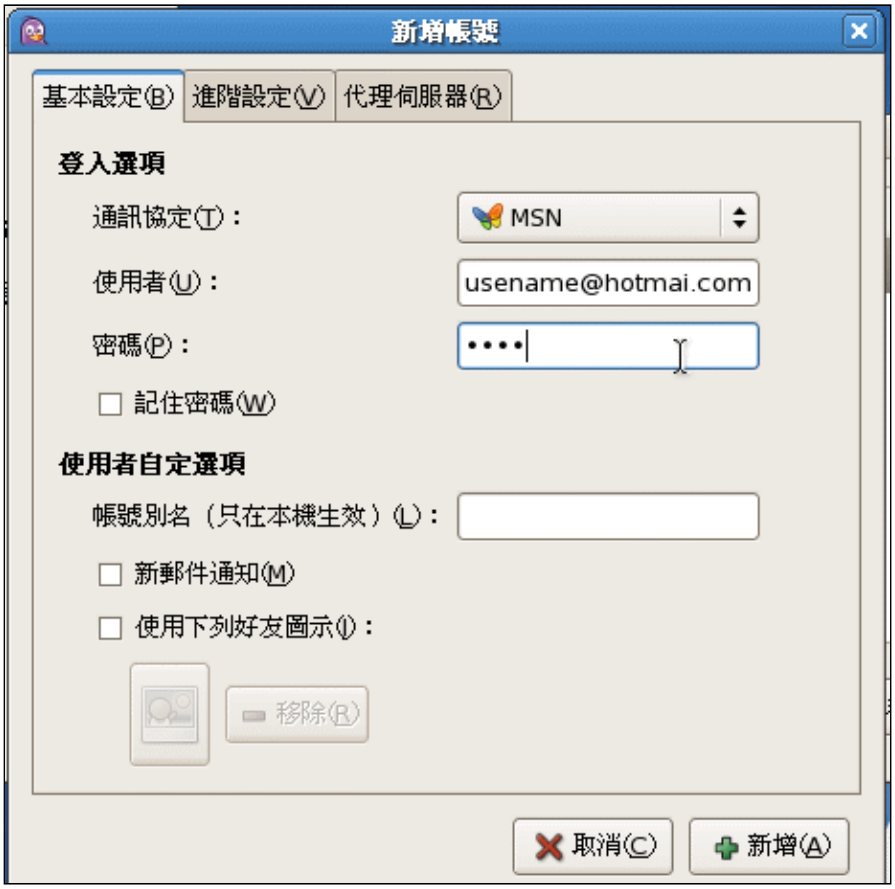

圖 5.3-3、設定 MSN 的帳號示意圖

如上圖，在畫面中輸入你的帳號與密碼，如果是在公用的電腦上，千萬不要按下『記住密碼』項目喔！按下新增後， pidgin 預設就會嘗試登入了！登入後的畫面如下所示：

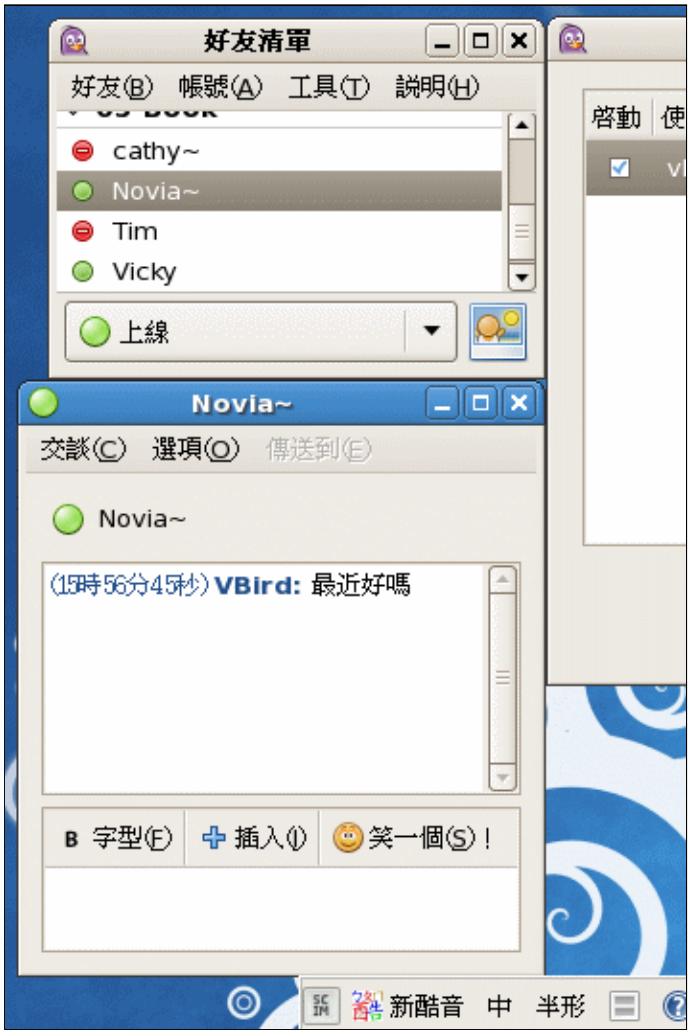

圖 5.3-4、使用 pidgin 的 MSN 方式進行連天囉

如果想要登出了，那麼就按下圖 5.3-4 最右邊那個視窗，將『啟動』的那個方框勾選取消，你就直接登出囉！

5.4 文字介面網頁瀏覽

什麼？文字界面竟然有瀏覽器！別逗了好不好？呵呵！誰有那個時間在逗你呦！真的啦！有這個東西， 是在文字界面下上網瀏覽的好工具！分別是 links 及 wget 這兩個寶貝蛋，但是，你必需要確定你已經安裝了這兩個套件才行。 好佳在的是，CentOS 預設這兩個玩意兒都有安裝喔！底下就讓我們來聊一聊這兩個好用的傢伙吧！

5.4.1 文字瀏覽器：links

其實早期鳥哥最常使用的是 lynx 這個文字瀏覽器，不過 CentOS 從 5.x 以後預設使用的文字瀏覽器是 links 這一支，這兩支的使用方式又非常的類似，因此，在這一版當中，我們就僅介紹 links 囉！若對 lynx 有興趣的話， 自己 man 一下吧！

這個指令可以讓我們來瀏覽網頁，但鳥哥認為，這個檔案最大的功能是在『查閱 Linux 本機上面以 HTML 語法寫成的文件資料 (document)』 怎麼說呢？如果你曾經到 Linux 本機底下的 /usr/share/doc 這個

目錄看過文件資料的話， 就會常常發現一些網頁檔案，使用 vi 去查閱時，老是看到一堆 HTML 的語法！有礙閱讀啊～ 這時候使用 links 就是個好方法啦！可以看的清清楚楚啊！ ^_^

```
[root@www ~]# links [options] [URL]
選項與參數：
-anonymous [0|1]：是否使用匿名登入的意思；
-dump [0|1]    ：是否將網頁的資料直接輸出到 standard out 而非 links 軟體功能
-dump_charset  ：後面接想要透過 dump 輸出到螢幕的語系編碼，big5 使用 cp950 喔

# 範例一：瀏覽 Linux kernel 網站
[root@www ~]# links http://www.kernel.org
```

當我直接輸入 links 網站網址後，就會出現如下的圖示：

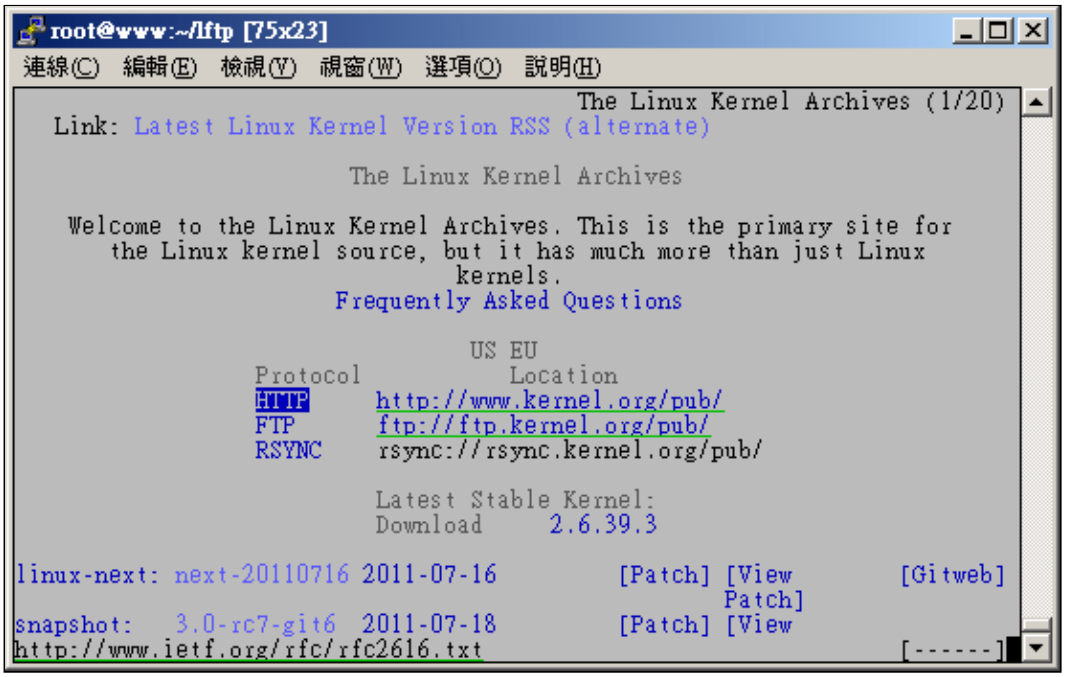

圖 5.4-1、使用 links 查詢網頁資料的顯示結果

上面這個畫面的基本說明如下：

- 進入畫面之後，由於是文字型態，所以編排可能會有點位移！不過不打緊！不會影響我們看咚咚！
- 這個時候可以使用『上下鍵』來讓游標在上面的選項當中(如信箱、書籤等等的)，按下 Enter 就進入該頁面
- 可以使用『左右鍵』來移動『上一頁或下一頁』
- 一些常見功能按鍵：
  - h：history ，曾經瀏覽過的 URL 就顯示到畫面中
  - g：Goto URL，按 g 後輸入網頁位址(URL) 如 :http://www.abc.edu/等
  - d：download，將該連結資料下載到本機成為檔案；
  - q：Quit，離開 links 這個軟體；
  - o：Option，進入功能參數的設定值修改中，最終可寫入 ~/.elinks/elinks.conf 中
  - Ctrl+C ：強迫切斷 links 的執行。
  - 方向鍵:
    - 上 ：移動游標至本頁中 "上一個可連結點" .
    - 下 ：移動游標至本頁中 "下一個可連結點" .
    - 左 ：back. 跳回上一頁.

- 右 ：進入反白游標所連結之網頁.
- ENTER 同滑鼠 "右" 鍵.

至於如果是瀏覽 Linux 本機上面的網頁檔案，那就可以使用如下的方式：

```
[root@www ~]# links /usr/share/doc/HTML/index.html
```

在鳥哥的 CentOS 6.x 當中，有這麼一個檔案，我就可以利用 links 來取出察看吶！顯示的結果有點像底下這樣：

圖 5.4-2、使用 links 查詢本機的 HTML 文件檔案

當然啦！因為你的環境可能是在 Linux 本機的 tty1~tty6 ，所以無法顯示出中文，這個時候你就得要設定為： 『LANG=en_US』之類的語系設定才行喔！另外，如果某些時刻你必須上網點選某個網站以自動取得更新時。 舉例來說，早期的自動線上更新主機名稱系統，僅支援網頁更新，那你如何進行更新呢？嘿嘿！可以使用 links 喔！利用 -dump 這個參數處理先：

```
# 透過 links 將 tw.yahoo.com 的網頁內容整個抓下來儲存
[root@www ~]# links -dump http://tw.yahoo.com > yahoo.html

# 某個網站透過 GET 功能可以上傳帳號為 user 密碼為 pw ，用文字介面處理為：
[root@www ~]# links -dump \
> http://some.site.name/web.php?name=user&password=pw > testfile
```

上面的網站後面有加個問號 (?) 對吧？後面接的則是利用網頁的『 GET 』功能取得的各項變數資料， 利用這個功能，我們就可以直接點選到該網站上囉！非常的方便吧！而且會將執行的結果輸出到 testfile 檔案中，不過如果網站提供的資料是以『 POST 』為主的話，那鳥哥就不知道如何搞定了。 GET 與 POST 是 WWW 通訊協定中，用來將資料透過瀏覽器上傳到伺服器端的一種方式， 一般來說，目前討論區或部落格等，大多使用可以支援較多資料的 POST 方式上傳啦！ 關於 GET 與 POST 的相關資訊我們會在第二十章 WWW 伺服器當中再次的提及！

5.4.2 文字介面下載器： wget

如果說 links 是在進行網頁的『瀏覽』，那麼 wget 就是在進行『網頁資料的取得』。舉例來說，我們的 Linux 核心是放置在 www.kernel.org 內，主要同時提供 ftp 與 http 來下載。我們知道可以使用 lftp 來下載資料，但如果想要用瀏覽器來下載呢？那就利用 wget 吧！

```
[root@www ~]# wget [option] [網址]
選項與參數：
若想要連線的網站有提供帳號與密碼的保護時，可以利用這兩個參數來輸入喔！
--http-user=usrname
--http-password=password
--quiet ：不要顯示 wget 在抓取資料時候的顯示訊息
更多的參數請自行參考 man wget 吧！^_^

# 範例一：請下載 2.6.39 版的核心
[root@www ~]# wget \
> http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.39.tar.bz2
--2011-07-18 16:58:26--  http://www.kernel.org/pub/linux/kernel/v2.6/..
Resolving www.kernel.org... 130.239.17.5, 149.20.4.69, 149.20.20.133, ...
Connecting to www.kernel.org|130.239.17.5|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 76096559 (73M) [application/x-bzip2]
Saving to: `linux-2.6.39.tar.bz2'

88% [================================>        ] 67,520,536  1.85M/s  eta 7s
```

你瞧瞧～很可愛吧！不必透過瀏覽器，只要知道網址後，立即可以進行檔案的下載， 又快速又方便，還可以透過 proxy 的幫助來下載呢！透過修改 /etc/wgetrc 來設定你的代理伺服器：

```
[root@www ~]# vim /etc/wgetrc
#http_proxy = http://proxy.yoyodyne.com:18023/   <==找到底下這幾行，大約在 78 行
#ftp_proxy = http://proxy.yoyodyne.com:18023/
#use_proxy = on

# 將他改成類似底下的模樣，記得，你必須要有可接受的 proxy 主機才行！
http_proxy = http://proxy.ksu.edu.tw:3128/
use_proxy = on
```

5.5 封包擷取功能

很多時候由於我們的網路連線出現問題，使用類似 ping 的軟體功能卻又無法找出問題點，最常見的是因為路由與 IP 轉遞後所產生的一些困擾 (請參考防火牆與 NAT 主機部分)，這個時候要怎麼辦？最簡單的方法就是『分析封包的流向』囉！透過分析封包的流向，我們可以瞭解一條連線應該是如何進行雙向的連線的動作， 也就會清楚的瞭解到可能發生的問題所在了！底下我們就來談一談這個 tcpdump 與圖形介面的封包分析軟體吧！

5.5.1 文字介面封包擷取器： tcpdump

說實在的，對於 tcpdump 這個軟體來說，你甚至可以說這個軟體其實就是個駭客軟體， 因為他不但可以分析封包的流向，連封包的內容也可以進行『監聽』， 如果你使用的傳輸資料是明碼的話，不得了，在

router 或 hub 上面就可能被人家監聽走了！ 我們在第二章談到的 CSMA/CD 流程中，不是說過有所謂的『監聽軟體』嗎？這個 tcpdump 就是啦！ 很可怕吶！所以，我們也要來瞭解一下這個軟體啊！(註：這個 tcpdump 必須使用 root 的身份執行)

```
[root@www ~]# tcpdump [-AennqX] [-i 介面] [-w 儲存檔名] [-c 次數] \
                [-r 檔案] [所欲擷取的封包資料格式]
選項與參數：
-A ：封包的內容以 ASCII 顯示，通常用來捉取 WWW 的網頁封包資料。
-e ：使用資料連接層 (OSI 第二層) 的 MAC 封包資料來顯示；
-nn：直接以 IP 及 port number 顯示，而非主機名與服務名稱
-q ：僅列出較為簡短的封包資訊，每一行的內容比較精簡
-X ：可以列出十六進位 (hex) 以及 ASCII 的封包內容，對於監聽封包內容很有用
-i ：後面接要『監聽』的網路介面，例如 eth0, lo, ppp0 等等的介面；
-w ：如果你要將監聽所得的封包資料儲存下來，用這個參數就對了！後面接檔名
-r ：從後面接的檔案將封包資料讀出來。那個『檔案』是已經存在的檔案，
   並且這個『檔案』是由 -w 所製作出來的。
-c ：監聽的封包數，如果沒有這個參數， tcpdump 會持續不斷的監聽，
   直到使用者輸入 [ctrl]-c 為止。
所欲擷取的封包資料格式：我們可以專門針對某些通訊協定或者是 IP 來源進行封包擷取，
   那就可以簡化輸出的結果，並取得最有用的資訊。常見的表示方法有：
   'host foo', 'host 127.0.0.1' ：針對單部主機來進行封包擷取
   'net 192.168' ：針對某個網域來進行封包的擷取；
   'src host 127.0.0.1' 'dst net 192.168'：同時加上來源(src)或目標(dst)限制
   'tcp port 21'：還可以針對通訊協定偵測，如 tcp, udp, arp, ether 等
   還可以利用 and 與 or 來進行封包資料的整合顯示呢！

# 範例一：以 IP 與 port number 捉下 eth0 這個網路卡上的封包，持續 3 秒
[root@www ~]# tcpdump -i eth0 -nn
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
17:01:47.360523 IP 192.168.1.101.1937 > 192.168.1.100.22: Flags [.], ack 196, win 65219,
17:01:47.362139 IP 192.168.1.100.22 > 192.168.1.101.1937: Flags [P.], seq 196:472, ack 1,
17:01:47.363201 IP 192.168.1.100.22 > 192.168.1.101.1937: Flags [P.], seq 472:636, ack 1,
17:01:47.363328 IP 192.168.1.101.1937 > 192.168.1.100.22: Flags [.], ack 636, win 64779,
<==按下 [ctrl]-c 之後結束
6680 packets captured                <==捉下來的封包數量
14250 packets received by filter     <==由過濾所得的總封包數量
7512 packets dropped by kernel       <==被核心所丟棄的封包
```

如果你是第一次看 tcpdump 的 man page 時，肯定一個頭兩個大，因為 tcpdump 幾乎都是分析封包的表頭資料，使用者如果沒有簡易的網路封包基礎，要看懂粉難吶！ 所以，至少你得要回到網路基礎裡面去將 TCP 封包的表頭資料理解理解才好啊！ ^_^！至於那個範例一所產生的輸出範例中，我們可以約略區分為數個欄位， 我們以範例一當中那個特殊字體行來說明一下：

- 17:01:47.362139：這個是此封包被擷取的時間，『時:分:秒』的單位；
- IP：透過的通訊協定是 IP ；
- 192.168.1.100.22 > ：傳送端是 192.168.1.100 這個 IP，而傳送的 port number 為 22，你必須要瞭解的是，那個大於 (>) 的符號指的是封包的傳輸方向喔！
- 192.168.1.101.1937：接收端的 IP 是 192.168.1.101， 且該主機開啟 port 1937 來接收；
- [P.], seq 196:472：這個封包帶有 PUSH 的資料傳輸標誌， 且傳輸的資料為整體資料的 196~472 byte；
- ack 1：ACK 的相關資料。

最簡單的說法，就是該封包是由 192.168.1.100 傳到 192.168.1.101，透過的 port 是由 22 到 1937 ， 使用的是 PUSH 的旗標，而不是 SYN 之類的主動連線標誌。呵呵！不容易看的懂吧！所以說，上頭才講請務必到 TCP 表頭資料的部分去瞧一瞧的啊！

再來，一個網路狀態很忙的主機上面，你想要取得某部主機對你連線的封包資料而已時， 使用 tcpdump 配合管線命令與正規表示法也可以，不過，畢竟不好捉取！ 我們可以透過 tcpdump 的表示法功能，就能夠輕易的將所需要的資料獨立的取出來。 在上面的範例一當中，我們僅針對 eth0 做監聽，所以整個 eth0 介面上面的資料都會被顯示到螢幕上， 不好分析啊！那麼我們可以簡化嗎？例如只取出 port 21 的連線封包，可以這樣做：

```
[root@www ~]# tcpdump -i eth0 -nn port 21
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
01:54:37.96 IP 192.168.1.101.1240 > 192.168.1.100.21: . ack 1 win 65535
01:54:37.96 IP 192.168.1.100.21 > 192.168.1.101.1240: P 1:21(20) ack 1 win 5840
01:54:38.12 IP 192.168.1.101.1240 > 192.168.1.100.21: . ack 21 win 65515
01:54:42.79 IP 192.168.1.101.1240 > 192.168.1.100.21: P 1:17(16) ack 21 win 65515
01:54:42.79 IP 192.168.1.100.21 > 192.168.1.101.1240: . ack 17 win 5840
01:54:42.79 IP 192.168.1.100.21 > 192.168.1.101.1240: P 21:55(34) ack 17 win 5840
```

瞧！這樣就僅提出 port 21 的資訊而已，且仔細看的話，你會發現封包的傳遞都是雙向的， client 端發出『要求』而 server 端則予以『回應』，所以，當然是有去有回啊！ 而我們也就可以經過這個封包的流向來瞭解到封包運作的過程。舉例來說：

1. 我們先在一個終端機視窗輸入『 tcpdump -i lo -nn 』 的監聽，
2. 再另開一個終端機視窗來對本機 (127.0.0.1) 登入『ssh localhost』

那麼輸出的結果會是如何？

```
[root@www ~]# tcpdump -i lo -nn
1 tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
2 listening on lo, link-type EN10MB (Ethernet), capture size 96 bytes
3 11:02:54.253777 IP 127.0.0.1.32936 > 127.0.0.1.22: S 933696132:933696132(0)
   win 32767 <mss 16396,sackOK,timestamp 236681316 0,nop,wscale 2>
4 11:02:54.253831 IP 127.0.0.1.22 > 127.0.0.1.32936: S 920046702:920046702(0)
   ack 933696133 win 32767 <mss 16396,sackOK,timestamp 236681316 236681316,nop,
   wscale 2>
5 11:02:54.253871 IP 127.0.0.1.32936 > 127.0.0.1.22: . ack 1 win 8192 <nop,
   nop,timestamp 236681316 236681316>
6 11:02:54.272124 IP 127.0.0.1.22 > 127.0.0.1.32936: P 1:23(22) ack 1 win 8192
   <nop,nop,timestamp 236681334 236681316>
7 11:02:54.272375 IP 127.0.0.1.32936 > 127.0.0.1.22: . ack 23 win 8192 <nop,
   nop,timestamp 236681334 236681334>
```

上表顯示的頭兩行是 tcpdump 的基本說明，然後：

- 第 3 行顯示的是『來自 client 端，帶有 SYN 主動連線的封包』，
- 第 4 行顯示的是『來自 server 端，除了回應 client 端之外(ACK)，還帶有 SYN 主動連線的標誌；
- 第 5 行則顯示 client 端回應 server 確定連線建立 (ACK)
- 第 6 行以後則開始進入資料傳輸的步驟。

從第 3-5 行的流程來看，熟不熟悉啊？沒錯！那就是三向交握的基礎流程啦！夠有趣吧！ 不過 tcpdump 之所以被稱為駭客軟體之一可不止上頭介紹的功能吶！ 上面介紹的功能可以用來作為我們主機的封包連

線與傳輸的流程分析， 這將有助於我們瞭解到封包的運作，同時瞭解到主機的防火牆設定規則是否有需要修訂的地方。

更神奇的使用要來啦！如果我們使用 tcpdump 在 router 上面監聽『明碼』的傳輸資料時， 例如 FTP 傳輸協定，你覺得會發生什麼問題呢？ 我們先在主機端下達『 tcpdump -i lo port 21 -nn -X 』然後再以 ftp 登入本機，並輸入帳號與密碼， 結果你就可以發現如下的狀況：

```
[root@www ~]# tcpdump -i lo -nn -X 'port 21'
    0x0000:  4500 0048 2a28 4000 4006 1286 7f00 0001  E..H*(@.@.......
    0x0010:  7f00 0001 0015 80ab 8355 2149 835c d825  ........U!I.\.%
    0x0020:  8018 2000 fe3c 0000 0101 080a 0e2e 0b67  .....<.........g
    0x0030:  0e2e 0b61 3232 3020 2876 7346 5450 6420  ...a220.(vsFTPd.
    0x0040:  322e 302e 3129 0d0a                      2.0.1)..

    0x0000:  4510 0041 d34b 4000 4006 6959 7f00 0001  E..A.K@.@.iY....
    0x0010:  7f00 0001 80ab 0015 835c d825 8355 215d  .........\.%.U!]
    0x0020:  8018 2000 fe35 0000 0101 080a 0e2e 1b37  .....5.........7
    0x0030:  0e2e 0b67 5553 4552 2064 6d74 7361 690d  ...gUSER.dmtsai.
    0x0040:  0a                                       .

    0x0000:  4510 004a d34f 4000 4006 694c 7f00 0001  E..J.O@.@.iL....
    0x0010:  7f00 0001 80ab 0015 835c d832 8355 217f  .........\.2.U!.
    0x0020:  8018 2000 fe3e 0000 0101 080a 0e2e 3227  .....>........2'
    0x0030:  0e2e 1b38 5041 5353 206d 7970 6173 7377  ...8PASS.mypassw
    0x0040:  6f72 6469 7379 6f75 0d0a                 ordisyou..
```

上面的輸出結果已經被簡化過了，你必須要自行在你的輸出結果當中搜尋相關的字串才行。 從上面輸出結果的特殊字體中，我們可以發現『該 FTP 軟體使用的是 vsftpd ，並且使用者輸入 dmtsai 這個帳號名稱，且密碼是 mypasswordisyou』 嘿嘿！你說可不可怕啊！如果使用的是明碼的方式來傳輸你的網路資料？ 所以我們才常常在講啊，網路是很不安全滴！

另外你得瞭解，為了讓網路介面可以讓 tcpdump 監聽，所以執行 tcpdump 時網路介面會啟動在 『錯亂模式 (promiscuous)』 ，所以你會在 /var/log/messages 裡面看到很多的警告訊息， 通知你說你的網路卡被設定成為錯亂模式！別擔心，那是正常的。至於更多的應用，請參考 man tcpdump 囉！

例題：

如何使用 tcpdump 監聽 (1)來自 eth0 介面卡且 (2)通訊協定為 port 22 ，(3)封包來源為 192.168.1.101 的封包資料？

答：

tcpdump -i eth0 -nn 'port 22 and src host 192.168.1.101'

5.5.2 圖形介面封包擷取器： wireshark

tcpdump 是文字介面的封包擷取器，那麼有沒有圖形介面的？有啊！那就是 wireshark (註1) 這套軟體。這套軟體早期稱為 ethereal ，目前同時提供文字介面的 tethereal 以及圖形介面的 wireshark 兩個咚咚。由於我們當初安裝時預設並沒有裝這套，因此妳必須要先使用 yum 去網路安裝喔！也可以拿出光碟來安裝啦！有兩套需要安裝，分別是文字介面的 wireshark 以及圖形介面的 wireshark-gnome 軟體。安裝方式如下：

```
[root@www ~]# yum install wireshark wireshark-gnome
```

啟動這套軟體的方法很簡單，你必須要在 X Window 底下，透過『應用程式』-->『網際網路』-->『wireshark network analyzer』就可以啟動啦！啟動的畫面如下所示：

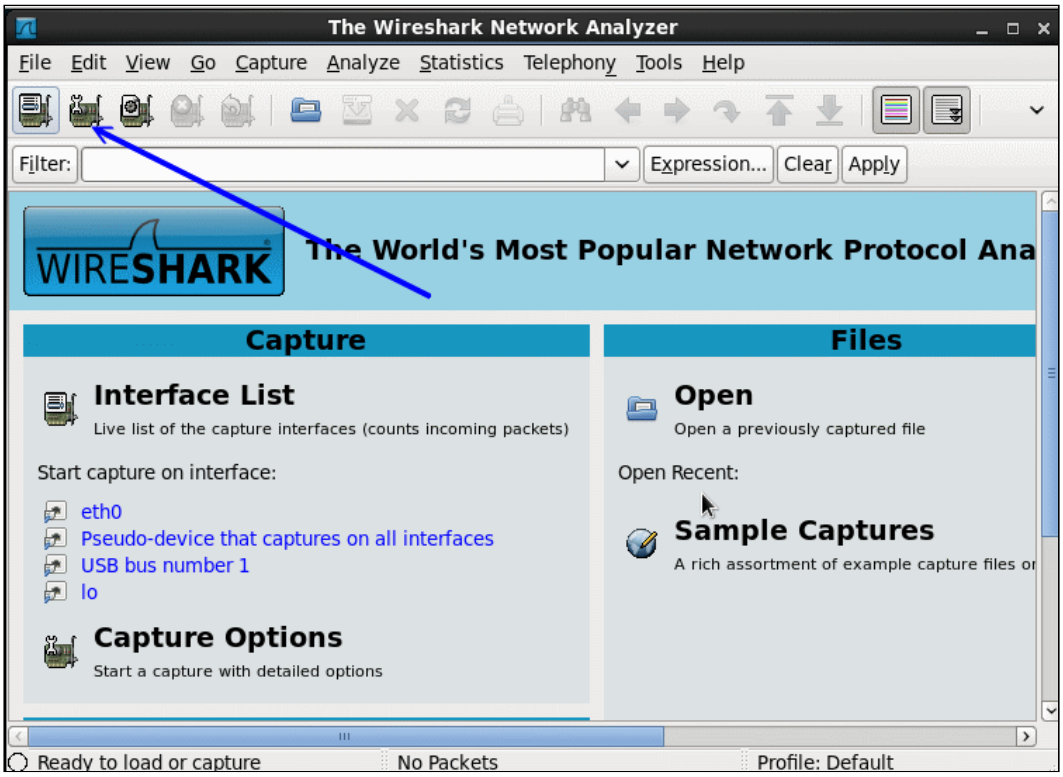

圖 5.5-1、wireshark 的使用示意圖

其實這一套軟體功能非常強大！鳥哥這裡僅講簡單的用法，若有特殊需求，就得要自己找找資料囉。 想要開始擷取封包前，得要設定一下監聽的介面之類的，因此點選圖 5.5-1 畫面中的網路卡小圖示吧！ 就會出現如下的畫面給你選擇了。

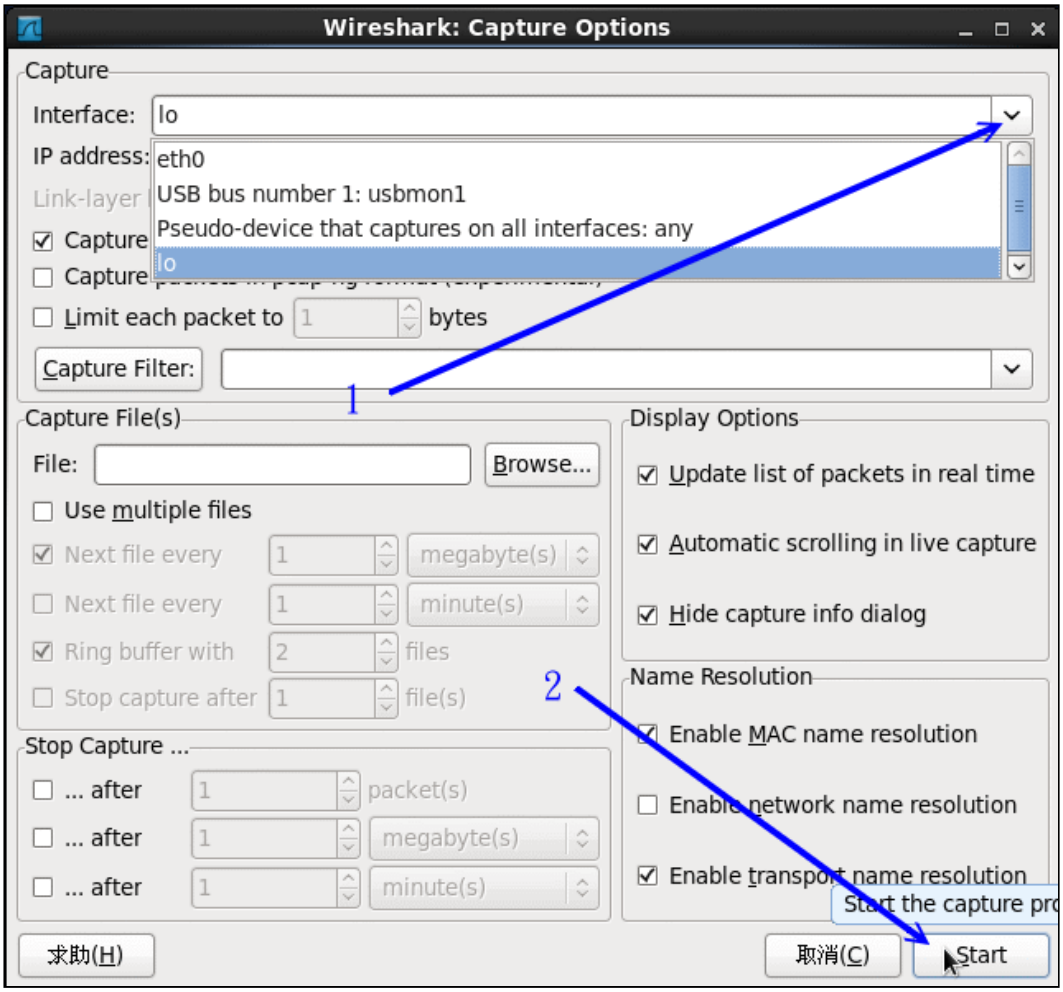

圖 5.5-2、wireshark 的使用示意圖

在上圖中，你得先選擇想要監聽的介面，鳥哥這裡因為擔心外部的封包太多導致畫面很亂，因此這裡使用內部的 lo 介面來作為範例。你得要注意， lo 平時是很安靜的！所以，鳥哥在點選了『start』之後，還有打開終端機， 之後使用『 ssh localhost 』來嘗試登入自己，這樣才能夠獲得封包喔！如下圖所示：

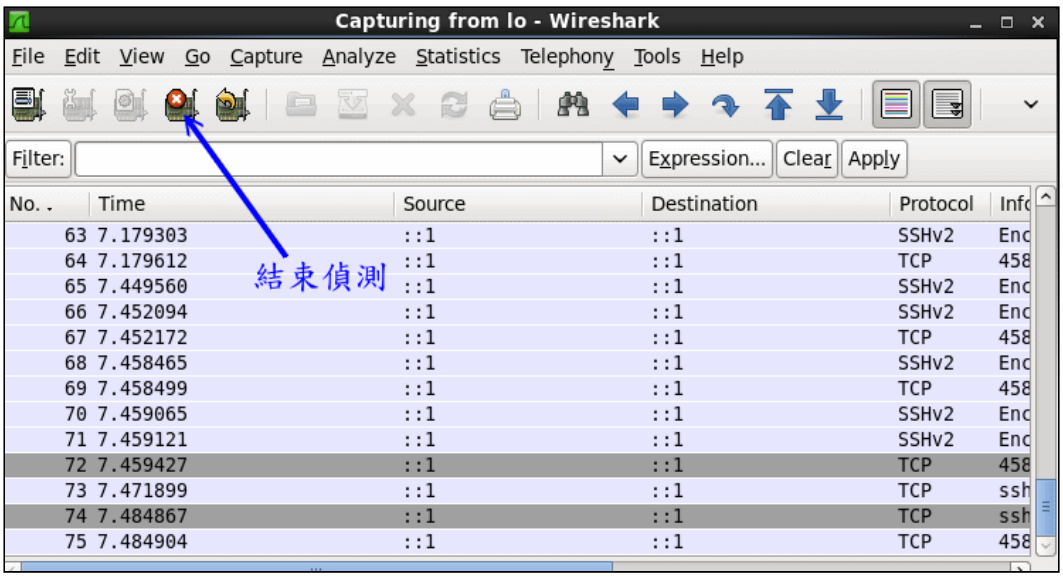

圖 5.5-3、wireshark 的使用示意圖

若沒有問題，等到你擷取了足夠的封包想要進行分析之後，按下圖 5.5-3 畫面中的停止小圖示，那麼封包擷取的動作就會終止， 接下來，就讓我們來開始分析一下封包吧！

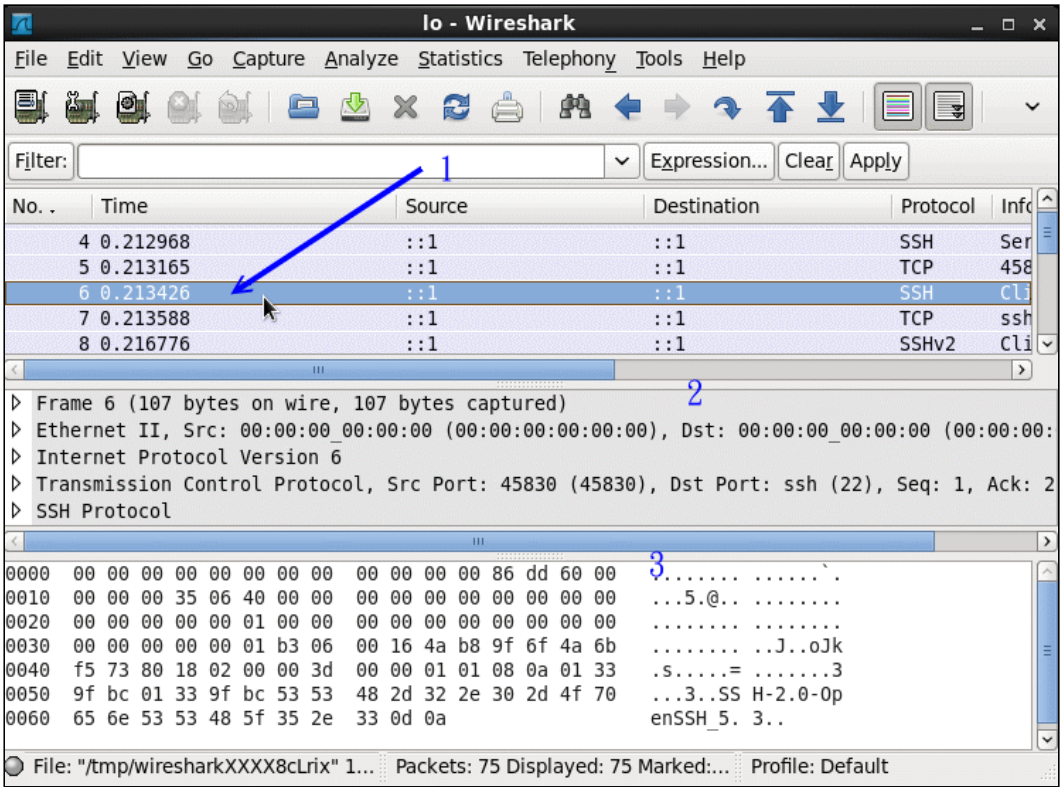

圖 5.5-4、wireshark 的使用示意圖

整個分析的畫面如上所示，畫面總共分為三大區塊，你可以將滑鼠游標移動到每個區塊中間的移動棒，就可以調整每個區塊的範圍大小了。第一區塊主要顯示的是封包的標頭資料，內容就有點類似 tcpdump 的顯示結果，第二區塊則是詳細的表頭資料，包括訊框的內容、通訊協定的內容以及 socket pair 等等資訊。 第三區塊則是 16 進位與 ASCII 碼的顯示結果 (詳細的封包內容)。

如果你覺得某個封包有問題，在畫面 1 的地方點選該封包 (圖例中是第 6 個封包)，那麼畫面 2 與 3 就會跟著變動！由於鳥哥測試的封包是加密資料的封包，因此畫面 2 顯示出封包表頭，但畫面 3 的封包內容就是亂碼啦！ 透過這個 wireshark 你就可以一口氣得到所需要的所有封包內容啦！而且還是圖形介面的，很方便吧！

5.5.3 任意啟動 TCP/UDP 封包的埠口連線： nc, netcat

這個 nc 指令可以用來作為某些服務的檢測，因為他可以連接到某個 port 來進行溝通，此外，還可以自行啟動一個 port 來傾聽其他用戶的連線吶！非常的不錯用！如果在編譯 nc 軟體的時候給予『GAPING_SECURITY_HOLE』參數的話，嘿嘿！ 這個軟體還可以用來取得用戶端的 bash 哩！可怕吧！我們的 CentOS 預設並沒有給予上面的參數， 所以我們不能夠用來作為駭客軟體～但是 nc 用來取代 telnet 也是個很棒的功能了！(有的系統將執行檔 nc 改名為 netcat 啦！)

```
[root@www ~]# nc [-u] [IP|host] [port]
[root@www ~]# nc -l [IP|host] [port]
選項與參數：
-l ：作為監聽之用，亦即開啟一個 port 來監聽用戶的連線；
-u ：不使用 TCP 而是使用 UDP 作為連線的封包狀態

# 範例一：與 telnet 類似，連接本地端的 port 25 查閱相關訊息
[root@www ~]# yum install nc
[root@www ~]# nc localhost 25
```

這個最簡單的功能與 telnet 幾乎一樣吧！可以去檢查某個服務啦！不過，更神奇的在後面， 我們可以建立兩個連線來傳訊喔！舉個例子來說，我們先在伺服器端啟動一個 port 來進行傾聽：

```
# 範例二：啟動一個 port 20000 來監聽使用者的連線要求
[root@www ~]# nc -l localhost 20000 &
[root@www ~]# netstat -tlunp | grep nc
tcp        0      0 ::1:20000           :::*        LISTEN      5433/nc
# 啟動一個 port 20000  在本機上！
```

接下來你再開另外一個終端機來看看，也利用 nc 來連線伺服器，並且輸入一些指令看看喔！

```
[root@www ~]# nc localhost 20000
   <==這裡可以開始輸入字串了！
```

此時，在用戶端我們可以打入一些字，你會發現在伺服器端會同時出現你輸入的字眼吶！ 如果你同時給予一些額外的參數，例如利用標準輸入與輸出 (stdout, stdin) 的話，那麼就可以透過這個連線來作很多事情了！ 當然 nc 的功能不只如此，你還可以發現很多的用途喔！ 請自行到你主機內的 /usr/share/doc/nc-1.84/scripts/ 目錄下看看這些 script ，有幫助的吶！ 不過，如果你需要額外的編譯出含有 GAPING_SECURITY_HOLE 功能， 以使兩端連線可以進行額外指令的執行時，就得要自行下載原始碼來編譯了！

5.6 重點回顧

- 修改網路介面的硬體相關參數，可以使用 ifconfig 這個指令，包括 MTU 等等；
- ifup 與 ifdown 其實只是 script ，在使用時，會主動去 /etc/sysconfig/network-scripts 下找到相對應的裝置設定檔，才能夠正確的啟動與關閉；
- 路由的修改與查閱可以使用 route 來查詢，此外， route 亦可進行新增、刪除路由的工作；
- ip 指令可以用來作為整個網路環境的設定，利用 ip link 可以修改『網路裝置的硬體相關功能』， 包括 MTU 與 MAC 等等，可以使用 ip address 修改 TCP/IP 方面的參數，包括 IP 以及網域參數等等， ip route 則可以修改路由！
- ping 主要是透過 ICMP 封包來進行網路環境的檢測工作，並且可以使用 ping 來查詢整體網域可接受最大的 MTU 值；
- 偵察每個節點的連線狀況，可以使用 traceroute 這個指令來追蹤！
- netstat 除了可以觀察本機的啟動介面外，還可以觀察 Unix socket 的傳統插槽介面資料；
- host 與 nslookup 預設都是透過 /etc/resolv.conf 內設定的 DNS 主機來進行主機名稱與 IP 的查詢；
- lftp 可以用來匿名登入遠端的 FTP 主機；
- links 主要的功能是『瀏覽』，包括本機上 HTML 語法的檔案， wget 則主要在用來下載 WWW 的資料；
- 擷取封包以分析封包的流向，可使用 tcpdump ，至於圖形介面的 wireshark 則可以進行更為詳細的解析。
- 透過 tcpdump 分析三向交握，以及分析明碼傳輸的資料，可發現網路加密的重要性。
- nc 可用來取代 telnet 進行某些服務埠口的檢測工作。

5.7 本章習題

- 暫時將你的 eth0 這張網路卡的 IP 設定為 192.168.1.100 ，如何進行？

  ifconfig eth0 192.168.1.100

- 我要增加一個路由規則，以 eth0 連接 192.168.100.100/24 這個網域，應該如何下達指令？

  route add -net 192.l68.100.0 netmask 255.255.255.0 dev eth0

- 我的網路停頓的很厲害，尤其是連接到 tw.yahoo.com 的時候，那麼我應該如何檢查那個環節出了問題？

  traceroute tw.yahoo.com

- 我發現我的 Linux 主機上面有個連線很怪異，想要將他斷線，應該如何進行？

  以 root 的身份進行『netstat -anp |more』查出該連線的 PID，然後以『 kill -9 PID 』踢掉該連線。

- 你如何知道 green.ev.ncku.edu.tw 這部主機的 IP ？

  方法很多，可以利用 host green.ev.ncku.edu.tw 或 dig green.ev.ncku.edu.tw 或 nslookup green.ev.ncku.edu.tw 等方法找出

- 請找出你的機器上面最適當的 MTU 應該是多少？

  請利用『ping -c 3 -M do -s MTU yourIP 』找出你的 IP 的 MTU 數值。 事實上，你還可以先以 ip 設定網路卡較大的 MTU 後，在進行上述的動作，才能夠找出網域內適合的 MTU。

- 如何在終端機介面上面進行 WWW 瀏覽？又該如何下載 WWW 上面提供的檔案？

  要瀏覽可以使用 links 或 lynx ，至於要下載則使用 wget 這個軟體。

- 在終端機介面中，如何連接 bbs.sayya.org 這個 BBS ？

  利用 telnet bbs.sayya.org 即可連接上

- 請自行以 tcpdump 觀察本機端的 ssh 連線時，三向交握的內容
- 請自行回答：為何使用明碼傳輸的網路連線資料較為危險？並自行以軟體將封包取出，並與同學討論封包的資訊
- 請自行至 Internet 下載 nc(netcat) 的原始碼，並且編譯成為具有 GAPING_SECURITY_HOLE 的參數，然後建立一條連線使用 -e /bin/bash 嘗試將本地端的 bash 丟給目的端執行 (特殊功能，可讓 client 取得來自主機的 bash)。

5.8 參考資料與延伸閱讀

- 註1：wireshark 的官網網址：http://www.wireshark.org/

2002/07/31：第一次完成日期！
2003/08/19：重新編排版面，加入 jmcce 的安裝以及 MTU 的相關說明
2003/08/20：加入課後練習去
2003/09/19：加入參考用解答咯！
2005/03/24：route 的指令參數寫錯了！已經訂正！
2006/07/24：將舊的文章移動到 此處
2006/07/24：拿掉相關性不高的 JMCCE 中文終端機； 將 Windows 系統的 MTU 檢測修改方法移除。 也拿掉 ncftp 的說明
2006/08/02：修改了很多部分，加入一些封包偵測的功能程式，tcpdump, nc 等指令！
2010/08/28：將舊的，基於 CentOS 4.x 所撰寫的文章放置於此處
2010/09/03：加入 links 取消 lynx，ethereal 改成 wireshark，gaim 改成 pidgin 了，nc 指令的用法跟前幾版有點不同。
2011/07/18：將基於 CentOS 5.x 的文章移動到此處
2011/07/18：將資料修訂為 CentOS 6.x 的模樣！不過 tcpdump 的變化不大，部分資料為 CentOS 5.x 的擷取示意！
