

第十六章 架設 Web Server

16.1 WWW、HTTP 與 Apache

16.1.1 WWW 與 HTTP 簡介

先來介紹 WWW (World Wide Web) 這個名詞，其翻成中文就是所謂的全球資訊網，它是 Internet 上很重要的一項網路服務，可以提供進一步的網路資訊交換。早期尚未發展出 WWW 時，就只能傳輸一些文字資料而已，對於其他像影像、圖片、聲音等，必須先將檔案編碼後才能傳輸；而到了發展出 WWW 之後，資料交換的過程就可以包含文字、影像、圖片、聲音等等多媒體的資訊，所以現在大家在 Internet 上才可以透過超連結看到很多美美的網頁，而提供網頁瀏覽服務的伺服器就被稱為 Web Server 或 WWW Server。

WWW 的文件是由 html 語法所組成，當在撰寫 html 時，可以為不同的資料型態前面加上不同的資料屬性標籤，這樣當 Client 端在接收網頁時，就可以針對不同的資料型態作解析，最後呈現在使用者面前的就是一個很漂亮的網頁。

再來談到的是 HTTP，在 WWW Server 與 Client 端瀏覽器之間用來彼此溝通的語言就是 HTTP 協定，其全名為 Hyper Text Transfer Protocol，所以不論是 Server 端或 Client 端在做資料傳輸時都要依照 HTTP 的標準來進行，而我們所要架設的 Web Server 當然也要支援此項標準。

綜合以上所述可以了解到：在 Client 端使用瀏覽器程式 (explorer、netscape、mozilla) 來存取 Web Server 所提供的 WWW 資料時，就是透過 HTTP 協定來做網頁資料的傳輸，而瀏覽器的主要作用就是在解析 html 的程式碼，最後才可以將解析後的結果呈現在瀏覽器的視窗之中。

另外附帶一提的就是 URL (Uniform Resource Locator) 這個名詞，它是表示 WWW 連結時的語法，比如我們輸入網址：<http://www.hinet.net/>，這就是一個 URL 了，其中 http 是指出資料傳輸時要使用的協定，www.hinet.net 就是 FQDN 囉。所以當 Client 端輸入這樣一個 URL 時，會先透過 DNS 來找出此 FQDN 所對應的 IP 位址，並使用 http 協定來與 WWW 主機的 80 port 做聯繫。一般來說，URL 最後有以 "/" 做結尾時，如 <http://192.168.1.111/test/>，那麼 test 是代表目錄之意；那沒有以 "/" 做結尾時，如 <http://192.168.1.111/test>，則 test 代表的是一個檔案。

16.1.2 Apache 簡介

Web Server 最出名也最被廣泛使用的一套軟體為 Apache。Apache 的前身為 NCSA (National Center for Supercomputing Applications)，而 NCSA 於 1995 年為排名第一的 Web Server 軟體，後來 NCSA 經過一連串的修正便產生了今日舉世聞名的 Apache。至於

Apache 這個名稱的由來，是因為當初的研發團隊是針對 NCSA 這個老牌的伺服器軟體不斷的修正而來：**a patchy server**，所以才取名為 Apache。又因為 Apache 這個名稱剛好與美國的一個原住民名稱相同，所以 Apache 的 Logo 就以彩色的羽毛來作為其標誌圖案。

1995 年年底，Apache 1.0 版正式推出，而目前較新的版本為 2.x 版。

16.2 安裝 Apache 套件及瀏覽套件內容

16.2.1 檢視 Apache 套件的安裝

```
suselinux:~ # rpm -qa | grep apache
apache2-2.0.49-27.8
apache2-prefork-2.0.49-27.8
```

如您尚未安裝的話，趕快把光碟拿出來，利用 YaST 工具將其裝上，這樣可解決套件相依性的問題。但如您是用 rpm 指令安裝的話，請記得先安裝 libapr0-2.0.49-27.8.i586.rpm，再來才是 apache2 及 apache2-prefork。

```
suselinux:~ # rpm -ivh libapr0-2.0.49-27.8.i586.rpm
suselinux:~ # rpm -ivh apache2-2.0.49-27.8.i586.rpm apache2-prefork-2.0.49-27.8.i586.rpm
```

另外還有個選擇性的套件 **apache2-example-pages**，它只是提供一些網頁範例檔案的套件而已，不裝也行。

16.2.2 查看套件內容

```
suselinux:~ # rpm -ql apache2

/etc/apache2/
# Apache 的一些相關組態檔都安置在這個目錄中。

/etc/apache2/default-server.conf
# 此檔包含了基本的 Apache 設定項目。另外也可將虛擬主機設定在這裡。

/etc/apache2/httpd.conf
# Apache 的主要設定檔。

/etc/apache2/listen.conf
# 設定 Apache 所 listen 的介面位址及 port number。
```

```
/etc/apache2/mod_mime-defaults.conf
# 設定預設語系的檔案。

/etc/apache2/mod_userdir.conf
# 有關使用者個人網頁的相關設定。

/etc/apache2/server-tuning.conf
# 這個檔案的設定參數，可以用來調整 Apache 的執行效能。當 Apache 需求量很大時才考慮修改。

/etc/apache2/uid.conf
# 設定以什麼身分來執行 Apache 的 daemon (/usr/sbin/httpd2-prefork)。

/etc/apache2/vhosts.d/
# 這是一個包含虛擬主機設定檔的目錄。

/etc/init.d/apache2
# 管理 Apache 服務的 script。

/usr/sbin/apache2ctl
# 也可用這個指令來管理 Apache。另外也可用來測試設定語法的正確性。

/usr/sbin/htpasswd2
# 設定使用者認證的指令。

/usr/sbin/rcapache2
# 連結至 /etc/init.d/apache2 的符號連結檔。

/var/log/apache2/
# 此目錄包含 Apache 的一些紀錄檔，如 error_log 及 access_log 等。
```

至於 `apache2-prefork` 套件裡，最重要的就是 `/usr/sbin/httpd2-prefork` 這個監聽 80 port 的 daemon。

設定下次開機時啟動 Apache 服務：

```
suselinux:~ # chkconfig apache2 35
```

16.3 Apache 的相關設定檔說明

Apache 的設定檔被安置在 `/etc/apache2` 目錄下，您可以切換至這個目錄並概略的瀏覽一下有哪些檔案存在。以下我們會針對幾個比較重要的設定檔來做說明。

16.3.1 設定 Apache

`/etc/apache2/httpd.conf`

這是 Apache 的主要設定檔，裡邊包含了 `include` 敘述（指出 Apache 相關設定檔的存放位置）及部分參數的設定。檔案內容共區分成三大部分：Global Environment、Main server configuration 及 Virtual server configuration。

```
suselinuX:~ # vi /etc/apache2/httpd.conf

### Global Environment ###

# run under this user/group id
Include /etc/apache2/uid.conf

# - how many server processes to start (server pool regulation)
# - usage of KeepAlive
Include /etc/apache2/server-tuning.conf

# IP addresses / ports to listen on
Include /etc/apache2/listen.conf
Include /etc/apache2/mod_mime-defaults.conf
# 以上每個檔案的功用，請參考前面套件內容的說明。

# use .htaccess files for overriding,
AccessFileName .htaccess
# and never show them
<Files ~ "\.ht">
    Order allow,deny
    Deny from all
</Files>
# 這裡是定義每個目錄下可以存在的存取控制檔案名稱為 .htaccess，且拒絕任何來源端做存取。

# List of resources to look for when the client requests a directory
DirectoryIndex index.html index.html.var
# 當 client 端在存取某個目錄而未指定檔名時，則會自動比對該目錄下的檔案名稱與 DirectoryIndex
# 所定義的檔名有沒有符合者（依序比對），如果有的話，就直接開啟此網頁檔案。
```

'Main' server configuration

```
# The directives in this section set up the values used by the 'main'  
# server, which responds to any requests that aren't handled by a  
# <VirtualHost> definition. These values also provide defaults for  
# any <VirtualHost> containers you may define later in the file.  
#  
# All of these directives may appear inside <VirtualHost> containers,  
# in which case these default settings will be overridden for the  
# virtual host being defined.
```

```
Include /etc/apache2/default-server.conf
```

Virtual server configuration

```
Include /etc/apache2/vhosts.d/*.conf
```

```
# 這裡是告訴我們虛擬主機可以設定在 vhosts.d 目錄中，且檔案名稱需取名為 xxx.conf 才行。
```

 **/etc/apache2/default-server.conf**

```
suselinux:~ # vi /etc/apache2/default-server.conf
```

```
ServerName www.paching.com.tw
```

```
# 設定您這台 Apache 的主機名稱囉。
```

```
DocumentRoot "/srv/www/htdocs"
```

```
# 定義網站的根目錄位置。
```

```
# 比如 Client 所輸入的 URL 為 http://192.168.1.111/，這時候就會至 /srv/www/htdocs/ 目錄中去尋找
```

```
# 有無與 DirectoryIndex 所定義的檔名相同，只要有符合者就直接開啟該網頁檔案（一般為
```

```
# index.html）。
```

```
# 那如果 URL 為 http://192.168.1.111/test/，則會至 /srv/www/htdocs/test/ 目錄中去搜尋。
```

```
<Directory "/srv/www/htdocs">
```

```
Options followsymlinks
```

```
AllowOverride None
```

```
Order allow,deny
```

```
Allow from all
```

```
</Directory>
```

```
# 這是有關於目錄相關權限的設定說明，每一個目錄的設定區段是以 < Directory "directory-path">
# 做開始，< /Directory> 做結束。至於設定項目，請接著看以下的說明。
# Options 之後可接的值有：
#     Indexes：一般而言，若該目錄內無 DirectoryIndex 所定義的檔名時，則會顯示此目錄的內容。
#     FollowSymlinks：允許在此目錄下的 Symbolic Link file，能連結至此目錄外的檔案或目錄。
#     ExecCGI：允許在此目錄下執行 CGI 程式。
#     MultiViews：可支援多國的語言版本。
#     All：開放所有功能。這是預設。
#     None：關閉所有功能。

# AllowOverride
# 我們可以在每個目錄底下去建立一個 .htaccess 的隱藏檔，並且可以允許讓這個檔案內的設定項目
# 凌駕在這裡的設定之上，不過您得先在該目錄區段內使用 AllowOverride 去指定才行。以下就先來
# 看看 AllowOverride 可指派給 .htaccess 去做設定的項目有哪些：
#     AuthConfig：允許在 .htaccess 中對目錄做認證方面的設定。關於認證目錄的做法，後面會說明。
#     Limit：允許在 .htaccess 中設定對來源端主機的存取限制。
#     Options：就是剛剛才介紹的 Options 囉。
#     All：只要在 .htaccess 中所做的任何設定，都會凌駕在這裡的設定之上。
#     None：AllowOverride 設定成 None，表示會忽略 .htaccess 這個檔案。

# Order allow,deny
# Allow from all
# 設定來源端存取的限制，此處的設定是表示所有來源端都可以做存取。
# 若設定順序為 allow,deny，則最後是以 deny 做為判定的依據。比方您想拒絕少數、允許多數來源
# 端存取，那麼可以這麼設定：
# order allow,deny
# allow from all
# deny from 192.168.1 paching.com.tw
# 相反地，若只想允許少數、拒絕多數來源端做存取，則設定如下：
# order deny,allow
# deny from all
# allow from 192.168.0 192.168.5

Alias /icons/ "/usr/share/apache2/icons/"
# 此處是有關於別名的設定。以這裡來說，icons 為別名，也就是 Client 端在對 icons 目錄做存取時，
# 實際上是對 /usr/share/apache2/icons 目錄做存取。比如 Client 端輸入 http://192.168.1.111/icons/
# 時，由於有為 icons 設定別名，因此會轉而向 /usr/share/apache2/icons/ 目錄作存取。假使沒有
# 為 icons 設定別名時，那麼就會到 /srv/www/htdocs 目錄下搜尋有無 icons 目錄。這個搜尋的優先
# 順序請一定要搞清楚。
```

```
<Directory "/usr/share/apache2/icons">
```

```
Options Indexes MultiViews
```

```
AllowOverride None
```

```
Order allow,deny
```

```
Allow from all
```

```
</Directory>
```

這裡使用了 indexes，那就表示當您對 icons 目錄作存取時，若該目錄中並無存在一個與

DirectoryIndex 所定義的檔名相同時，就會顯示 icons 目錄的內容。

```
ScriptAlias /cgi-bin/ "/srv/www/cgi-bin/"
```

這裡別名的設定功能跟剛剛介紹的 Alias 差不多，但是目錄下存放的檔案類型不同。如果您是使用

ScriptAlias 的話，那表示可以在 /srv/www/cgi-bin 目錄中存放著一些 cgi 程式。簡單的說，cgi-bin 就

是放置 server scripts 的目錄所在，像 Openwebmail (網路郵局) 便會使用到該項別名的設定。

```
<Directory "/srv/www/cgi-bin">
```

```
AllowOverride None
```

```
Options +ExecCGI -Includes
```

```
Order allow,deny
```

```
Allow from all
```

```
</Directory>
```

/etc/apache2/listen.conf

```
suselinux:~ # vi /etc/apache2/listen.conf
```

```
Listen 80
```

這是設定 Apache 所監聽的介面位址及 port number。而這裡的設定是表示允許在所有的介面位址去

監聽 80 port。如果您有兩個介面，但只允許其中一個介面來監聽，則可設定如下：

```
# Listen 192.168.1.111:80
```

如不打算使用這個 well-known port 的話，請逕行修改；比如改成 "Listen 8000" 之後，那 Client

端就需在 URL 上去指定 8000 port，例: http://192.168.1.111:8000/

/etc/apache2/mod_mime-defaults.conf

這是設定預設語系編碼的地方。如果 Client 端在瀏覽網頁時，無法看到繁體中文而是呈現亂碼的話，那麼就修改一下囉：

```
suselinux:~ # vi /etc/apache2/mod_mime-defaults.conf
```

```
AddDefaultCharset Big5
```

```
# 把這個參數修改成 Big5 即可。
```

/etc/apache2/server-tuning.conf

```
suselinux:~ # vi /etc/apache2/server-tuning.conf
```

```
<IfModule prefork.c>
```

```
# 表示載入 prefork.c 這個 MPM (multiprocessing module) 模組。
```

```
StartServers 5
```

```
# Apache 啟動時的服務行程數為 5 個。
```

```
MinSpareServers 5
```

```
# 最少需保持 5 個閒置行程數。
```

```
MaxSpareServers 10
```

```
# 設定最大的閒置行程數為 10 個。
```

```
ServerLimit 150
```

```
# 限定 MaxClients 的上限。
```

```
MaxClients 150
```

```
# 設定最多能同時應付 150 個 Client 端的需求，即 Server 所能產生的最大行程數為 150。
```

```
MaxRequestsPerChild 0
```

```
# 當 "Keep Alive" 設定成 off 時，每個子行程所能產生的最大需求數。
```

```
</IfModule>
```

```
KeepAlive On
```

```
# 打開持續連線的功能。
```

```
# 當 Keep Alive 設定成 off 的情況下，一個 connection 就是一個 request，也就是說當 Client 完成
```

```
# 一個連線請求後，Server 就會馬上切斷連線；那現在要是開啟持續連線功能的話，在 Client 完成連
```

```
# 線請求後，Server 並不會馬上斷線，而是要等到所限定的時間到達時才會切斷連線。
```

```
MaxKeepAliveRequests 100
```

```
# 設定持續連線下的最大需求數。
```

KeepAliveTimeout 15

```
# 在持續連線下，Client 完成連線請求後，若 15 秒內沒有再提出任何的 request，則在 15 秒時間到  
# 達時，Server 就會斷開連線。
```

/etc/apache2/uid.conf

```
User wwwrun
```

```
Group www
```

```
# Apache 服務啟動時，是以 root 的身分來啟動主要控制程式 httpd2-prefork，然後再以 non-root id  
# 的身分去執行其下的子行程，而此 non-root id 的 user 即是這裡所設定的 wwwrun，其所屬的  
# default group 為 www。
```

/etc/apache2/mod_userdir.conf

這是有關使用者個人網頁的相關設定，包括個人網頁的根目錄位置、存取權限的設定等。

```
suslinux:~ # vi /etc/apache2/mod_userdir.conf
```

```
<IfModule mod_userdir.c>
```

```
    UserDir disabled root
```

```
    # 不開放存取 root 的個人網頁。
```

```
    # 如果您只是不允許 Client 瀏覽 root、barry、mary 的網頁，那麼可以這樣設定：
```

```
    # UserDir disabled root barry mary
```

```
    # 相反過來，如果只允許瀏覽 barry 及 mary 的網頁，其他都拒絕，則請設定如下：
```

```
    # UserDir disabled
```

```
    # UserDir enabled barry mary
```

```
    # 最後談到開放 root 個人網頁的做法：
```

```
    # 1. 將 " UserDir disabled root " 註解起來。
```

```
    # 2. 對 /root/public_html 這個目錄區段做相關存取權限的設定。
```

```
    # <Directory /root/public_html>
```

```
    #     Order allow,deny
```

```
    #     Allow from all
```

```
    # </Directory>
```

```
    # 3. 確定 /root 及 /root/public_html 對於 others 具有可執行 (x) 的權限。
```

```
# UserDir public_html
# 設定使用者個人網頁所存放的根目錄名稱，預設為 public_html。意思就是說個人網頁必須存
# 放在自己家目錄下的 public_html 目錄之中。
# 當 Client 端要瀏覽某個使用者的個人網頁時，必須在 URL 的路徑中使用 "~username" 去
# 指定，如果以存取使用者 barry 的網頁來說，則 Client 需輸入網址：
# http://192.168.1.111/~barry/，這樣才能對 /home/barry/public_html/ 做存取。
# 當然這個根目錄名稱可以做修改，比如由 public_html 改成 userweb 後，那麼家目錄中就必
# 須存在 userweb 這個目錄，另外您還需對這個目錄做存取權限方面的設定，只要把底下關
# 於 <Directory /home/*/public_html> 中的 public_html 改成 userweb 即可。
```

```
<Directory /home/*/public_html>
```

```
AllowOverride FileInfo AuthConfig Limit Indexes
Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
```

```
<Limit GET POST OPTIONS PROPFIND>
```

```
Order allow,deny
```

```
Allow from all
```

```
</Limit>
```

```
<LimitExcept GET POST OPTIONS PROPFIND>
```

```
Order deny,allow
```

```
Deny from all
```

```
</LimitExcept>
```

```
</Directory>
```

```
</IfModule>
```

```
# 這裡解釋一下 Limit 的用法，Limit 主要是對來源端的存取動作做進一步限定，比如 GET, POST, PUT,
# DELETE, CONNECT, OPTIONS, PATCH,...等等。舉個例子，假使您想拒絕 192.168.2.0/24 的來源端
# 瀏覽網頁，並只允許 192.168.3.0/24 的來源端張貼資料，那麼可設定如下：
```

```
# <Limit GET>
```

```
# Order allow,deny
```

```
# Allow from all
```

```
# Deny from 192.168.2
```

```
# </Limit>
```

```
#
```

```
# <Limit POST>
```

```
# Order deny,allow
```

```
# Deny from all
```

```
# Allow from 192.168.3.0/24
# </Limit>
```

設定完成後，第一次啟動 Apache，請使用以下任何一種方式來執行：

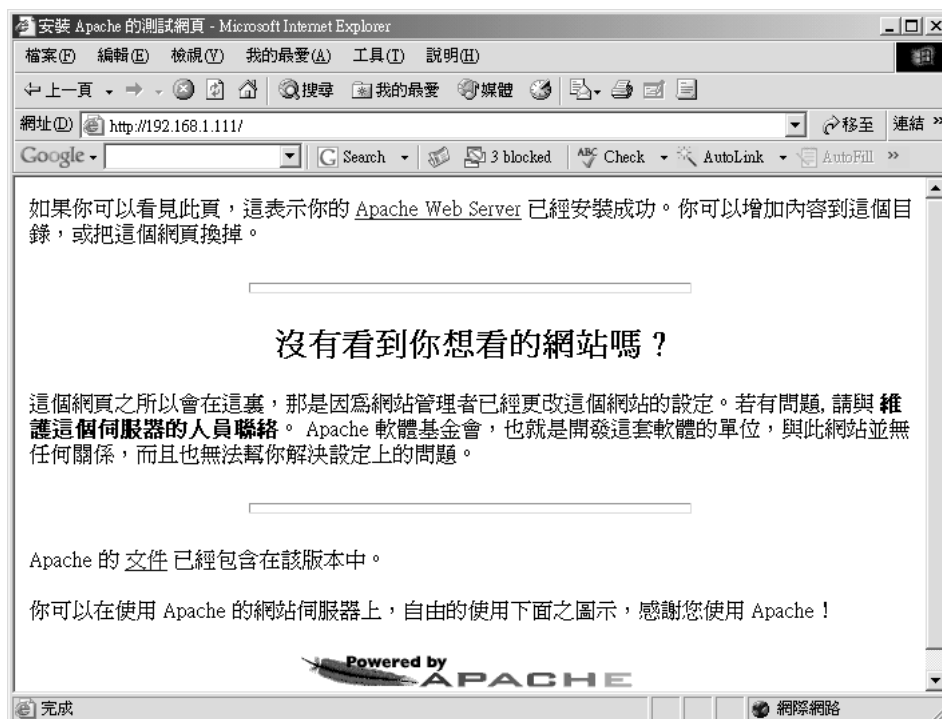
```
suselinu:~ # /etc/init.d/apache2 start
suselinu:~ # rcapache2 start
suselinu:~ # apache2ctl start
```

另外您也可以測試一下設定語法有無問題：

```
suselinu:~ # apache2ctl configtest
Syntax OK
suselinu:~ # httpd2 -t
Syntax OK
```

► Client 端測試：

這裡我們是以 Windows 的 Client 端來存取 192.168.1.111 這台 Apache 主機：



💡 如果您看不到這個 Home page 的話，先檢查看看 apache2-example-pages 套件有

沒有裝，不然就請直接在 /srv/www/htdocs 目錄下去建立一個 index.html 檔案。

16.3.2 建立使用者個人網頁

還記得 /etc/apache2/mod_userdir.conf 這個檔案吧，忘了的話，趕快再回過頭去稍作複習。

這裡我們以建立使用者 barry 的個人網頁為例，來看看要怎麼做：

1. 確定 public_html 目錄有存在於 barry 家目錄下，如不存在，請自行建立：

```
barry@suselinux:~> ls
Documents bin public_html
```

2. 建立 index.html 檔案於 public_html 目錄中：

```
barry@suselinux:~/public_html> cat > index.html
```

```
Welcome to barry's Web site.
```

3. 檢查相關目錄的權限：

```
barry@suselinux:~> ls -ld ; ls -ld public_html
drwxr-xr-x 7 barry users 560 Nov 17 04:46 .
drwxr-xr-x 2 barry users 144 Nov 17 04:54 public_html
```

→ 請務必確定家目錄及 public_html 具有給 others 可執行 (x) 的權限。

完成後，請回到 Client 端測試看看：



在存取個人網頁時，都要在 username 之前加上個 "~" 符號，如果這樣您嫌麻煩，而想直接輸入 username 就可以開啟網頁的話，那麼提供兩種做法給您參考（以 barry 為例）：

1. 設定 barry 的別名：

```
suselinux:/etc/apache2 # vi default-server.conf

Alias /barry/ "/home/barry/public_html/"

suselinux:~ # rcpapache2 restart
```

這樣當您所輸入的 URL 為 "http://192.168.1.111/barry/" 時，就會轉而向 /home/barry/public_html 做存取。

2. 建立符號連結檔：

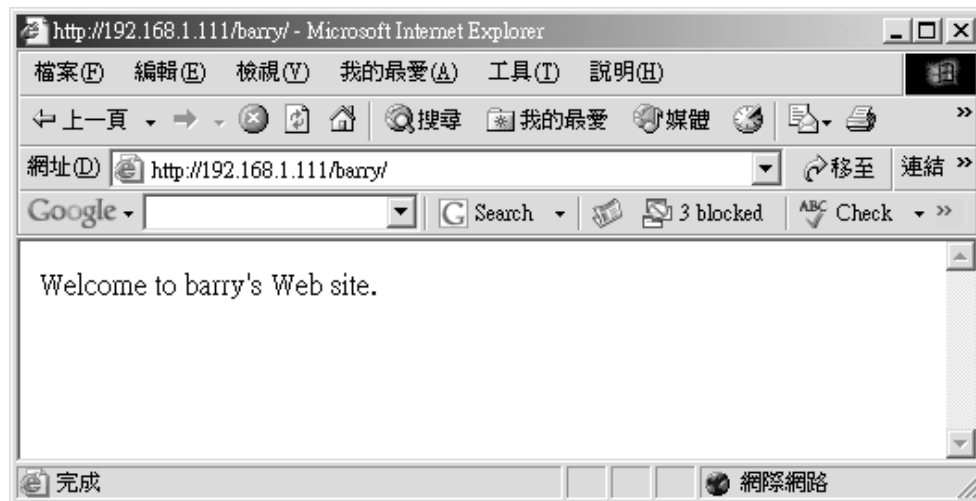
```
suselinux:/srv/www/htdocs # ln -s /home/barry/public_html barry

suselinux:/etc/apache2 # vi default-server.conf

<Directory "/srv/www/htdocs">
    Options FollowSymLinks
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>
```

Options 的部分請修改成 FollowSymLinks，這樣才能讓 /srv/www/htdocs/barry 這個符號連結檔能夠連結到此目錄外的地方。

測試：



16.4 設定認證目錄及 .htaccess 的使用

16.4.1 認證目錄

有時候一些比較特殊的網頁，您不希望每個人都可以存取，這時候可以將這些網頁檔案放置在指定的目錄之中，然後對此目錄做認證方面的設定，這樣當使用者在網頁中點選該連結的時候，就會跳出一個認證的視窗來要求您輸入帳號密碼，等認證通過以後，方可存取。

假使現在要做一個認證目錄叫 specialdir，接著就來看看要如何做相關的設定：

1. 先建立認證目錄：

```
suselinux:/srv/www/htdocs # mkdir specialdir
suselinux:/srv/www/htdocs # cat > specialdir/index.html
Congratulations on passing this authentication.
```

2. 設定認證目錄：

```
suselinux:~ # vi /etc/apache2/default-server.conf

<Directory "/srv/www/htdocs/specialdir">
    AuthName "It is top secret"
    # 這只是宣告認證的領域 (範圍) 而已。
    AuthType basic
    # 設定認證的類型。目前有 basic 及 Digest 可供使用。
    AuthUserFile "/etc/authfile"
```

指定使用者做認證時，所要依據的認證檔案。

require valid-user

這行設定是說，在認證檔案中的使用者，都是有效的使用者。當然您也可以指定只允許哪些人來

做認證，比如 "require user barry mary"，這樣就只有 barry 及 mary 可以對認證目錄做存取。

</Directory>

3. 重新啟動 Apache :

```
suselinux:~ # rcapache2 restart
```

4. 建立使用者認證資料 :

```
suselinux:~ # htpasswd2 -c /etc/authfile david
```

New password:

Re-type new password:

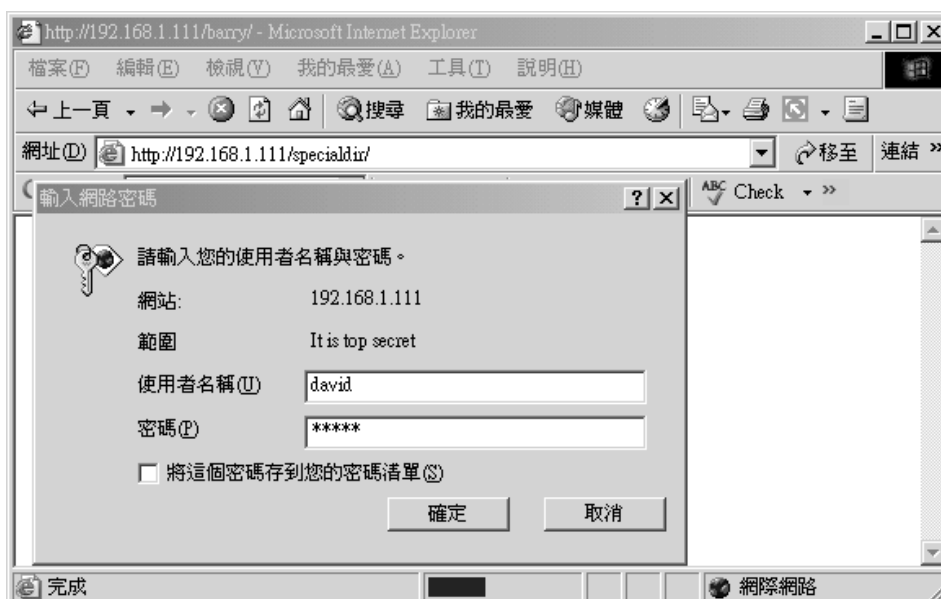
Adding password for user david

說明：

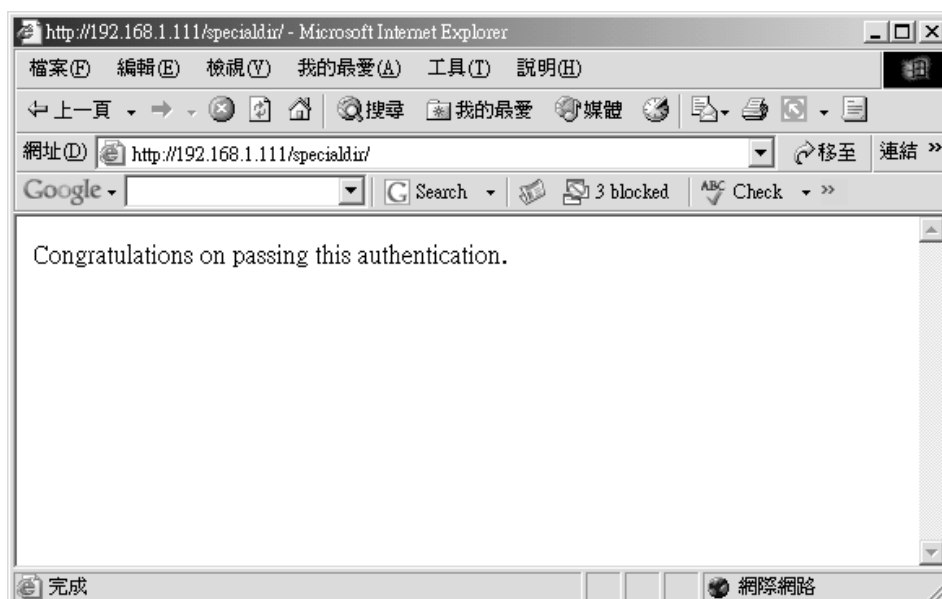
如 /etc/authfile 原本不存在，可使用 "-c" (create) 來將其建立起來；但如果這個檔案原本已存在，則不需加上此參數，萬一不小心加了，反而會覆蓋掉原先的認證資料，這點請注意。另外要刪除某個使用者的認證資料，則把 "-c" 參數改成 "-D" 即可，不然也可以直接編輯 /etc/authfile，將該使用者的資料直接移除。

► Client 端測試：

請於 URL 上輸入 <http://192.168.1.111/specialdir/>：



認證通過後，就會看到網頁的內容啦：



16.4.2 .htaccess 的使用

在 16.3.1 小節中提及 `/etc/apache2/default-server.conf` 時，於介紹 `AllowOverride` 中有提到 `.htaccess` 的功用，不記得的話，趕緊回去再複習一下。由那個地方的說明不難看出 `.htaccess` 必須要靠 `AllowOverride` 的指派，才能讓目錄下的 `.htaccess` 所設定的項目能夠凌駕在 Apache 設定檔中該目錄區段的設定項目之上。說到這裡，可能您還不是很明白什麼時候會使用到 `.htaccess`，舉個例子好了，假使今天有個使用者 `barry` 要求管理者幫其做個人網頁認證方面的設定，或要求對來源端的存取做進一步的控管等等，那此時管理者需做如下的設定：

```
suselinux:~ # vi /etc/apache2/mod_userdir.conf
```

```
<Directory "/home/barry/public_html">
    Options FollowSymLinks
    AuthName "Barry's Web Site"
    AuthType basic
    AuthUserFile "/etc/barryauth"
    require valid-user
    Order allow,deny
    Allow from all
    Deny from 192.168.2.0/24 domain.com.tw
</Directory>
```

```
suselinux:~ # rcapache2 restart
```

接著幫 barry 建立其所提供使用者的認證資料，以 user01 做代表：

```
suselinux:~ # htpasswd2 -c /etc/barryauth user01
New password:
Re-type new password:
Adding password for user user01
```

這樣就搞定使用者 barry 的要求了。但試想一下，如果今天有好幾十個使用者都向您提出同樣的要求，且每個使用者所提供給您的認證名單都不在少數的話，那這下子可就有得忙了。要解決您這個難題，AllowOverride 就派得上用場囉。

管理者只需在設定檔內做好 AllowOverride 的相關設定，就可以 [授權] 給每個使用者在目錄之中自行建立起 .htaccess 這個檔案，這時候使用者對此目錄的相關控管項目，就能夠依照自己的需求去做相關的調整。所以簡單的說，管理者先搞定 AllowOverride，然後再讓使用者自己去建立 .htaccess，並於該檔案內依照使用者自己的要求來做設定，而不必靠管理者幫他們一個個處理，如此便減輕很多管理者的負擔。詳細做法，請參考以下的範例說明。

 管理者的設定：

```
suselinux:~ # vi /etc/apache2/mod_userdir.conf

<Directory /home/*/public_html>
    AllowOverride  AuthConfig  Limit  options
    Order allow,deny
    Allow from all
</Directory>

suselinux:~ # rcapache2 restart
```

說明：

這個目錄區段的設定，是要讓一般使用者能夠在自己家目錄下的 public_html 中，去自行設定相關控管項目於 .htaccess 檔案內，比如使用者認證的設定、來源端主機的限制及 options 項目等。如果您想把所有設定項目，全部授權給使用者自己搞定的話，可設定 "AllowOverride All"，那如果只允許給使用者做認證方面設定的話，則只需指定 "AllowOverride AuthConfig"。



使用者的設定：

```
barry@suselinux:~/public_html> vi .htaccess

Options FollowSymLinks Indexes

AuthName "Barry's Web Site"
AuthType basic
AuthUserFile "/home/barry/authfile"
require user tom tina

<Limit GET>
    Order allow,deny
    Allow from all
    Deny from 192.168.2 192.168.5
</Limit>
<Limit POST>
    Order deny,allow
    Deny from all
</Limit>

barry@suselinux:~> /usr/sbin/htpasswd2 -c authfile tom
New password:
Re-type new password:
Adding password for user tom

barry@suselinux:~> /usr/sbin/htpasswd2 authfile tina
New password:
Re-type new password:
Adding password for user tina
```

都設定 OK 之後，不要忘了測試一下噢。

16.5 虛擬主機的設定

16.5.1 虛擬主機概述

一般來說，當 Client 端輸入不同的網址時，可以個別存取不同主機所提供的網站服務，那當然所瀏覽到的網頁內容都是不一樣的。今天如果提供網頁服務的主機只有一台，而當 Internet 的使用者輸入不同的網址時都是指到這台主機，並且開啟不同的網頁內容，也就是說主機上存在著一個以上的網站，此時站在使用者的角度來看，會以為這些網頁分別存在於不同主機

上面，但實際上卻是指向同一部主機，至於如何辦到的，這就是我們現在要探討的虛擬主機的設定了。

不過這裡有個附帶條件，就是今天您不管輸入什麼 URL，在 Internet 上必須要能解析得出此 FQDN 所對應的 IP 位址才行，不管是請別人幫您代指或者自己架設 DNS 都行，反正只要能解析得出來都可以。

一般我們在架設虛擬主機時，主要有兩種類型：

- **IP-based Virtual Hosts :**

此為早期在 Apache 上所設定的虛擬主機方式，每個虛擬主機名稱都有其各自的 IP Address，可能的實施方式就是一台主機放多片網卡，或者用 IP Alias 的做法綁多個 IP，但是不管怎麼做都還是要申請幾個 public address 才行，那萬一要是 IP 不足的話，可就壞了大事，於是後來才又發展出 Name-based Virtual Hosts。

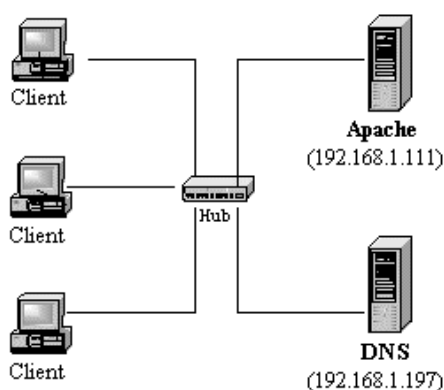
- **Name-based Virtual Hosts :**

將所有的虛擬主機都架設在同一個 IP Address，也就是把那些虛擬主機名稱都對應到同一個 IP 位址啦。比如我有一個合法的 IP 為 1.2.3.4，並且擁有兩個主機名稱分別為 barry.paching.com.tw 及 mary.pachng.com.tw，那麼這時候就需要讓這兩個 FQDN 在 Internet 上都能解析得出是對應到 1.2.3.4，這樣就是以名稱為基礎的虛擬主機囉。

16.5.2 設定虛擬主機

以下的設定範例是以 Name-based 為基礎來設定的。如目前您還未申請 domain name 的話也沒關係，可以自行在區網內模擬出這樣的環境，而待會兒的介紹就是以區網的環境來說明，以方便讀者自我練習。

以我的網路架構來說，Apache 主機的位址是 192.168.1.111，DNS 為 192.168.1.197，其他都是屬於 Windows 的 Client 端：



再來是規劃 DNS 所管理的 zone 為 paching.com.tw，而網站的 FQDN 為

www.paching.com.tw 及 virtual.paching.com.tw，因此在 DNS 的 zone file 內必須設定這兩個 FQDN 所對應的是同一個位址，也就是那台 Apache 的 192.168.1.111。最後就是 Client 端要指定其所使用的預設名稱伺服器為 192.168.1.197。

所以這個地方要分三個部分來做設定，分別是 Apache 虛擬主機的設定、DNS 名稱解析的設定及 Client 端的設定及測試。以下我們就分別來看看要怎麼做：

1. 設定虛擬主機 (192.168.1.111)：

(1) 建立另一個網站的根目錄及首頁 (Home Page)：

```
Apache:~ # mkdir /srv/www/virtualdir
Apache:~ # cd /srv/www/virtualdir
suselinux:/srv/www/virtualdir # cat > index.html ← 建立 Home Page。
It's virtual host.
```

(2) 設定 listen.conf：

```
Apache:~ # vi /etc/apache2/listen.conf

NameVirtualHost 192.168.1.111:80
# 這是設定虛擬主機所監聽的位址及使用的 port。
```

(3) 再來到 vhosts.d 目錄下建立 virtual.conf 檔案，並作如下的設定：

```
Apache:~ # cd /etc/apache2/vhosts.d
Apache:/etc/apache2/vhosts.d # vi virtual.conf

<VirtualHost 192.168.1.111:80>
    DocumentRoot "/srv/www/htdocs"
    ServerName www.paching.com.tw:80
</VirtualHost>

# 上面的設定是說，當 Client 端在 URL 中所輸入的主機名稱是 www.paching.com.tw 時，
# 就會對 /srv/www/htdocs 做存取。
# 另外要注意的是啟用虛擬主機功能時，如果沒有把原先的主機設定進來，反而會造成無法
# 存取 /srv/www/htdocs。

<VirtualHost 192.168.1.111:80>
    DocumentRoot "/srv/www/virtualdir"
    ServerName virtual.paching.com.tw:80
    <Directory "/srv/www/virtualdir">
```

```

        Order allow,deny
        Allow from all
    </Directory>
    # 由於 /srv/www/virtualdir 這個目錄在 Apache 的設定檔中，都沒有設定過有關存取權限
    # 方面的控管，因此這個地方務必要做設定。
</VirtualHost>

```

(4) 重新啟動 Apache：

```
Apache:~ # rcapache2 restart
```

2. 設定 DNS (192.168.1.197)：

(1) 設定 named.conf：

```

DNS:~ # vi /etc/named.conf
// 補上以下這個 zone 的宣告即可。

zone "paching.com.tw" in {
    type master;
    file "paching.db";
};

```

(2) 設定 zone file：

```

DNS:~ # vi /var/lib/named/paching.db

$TTL 43200
@           IN      SOA      dns.paching.com.tw.  barry.paching.com.tw.  (
                2005092801 10800 2700 864000 86400 )
           IN      NS       dns.paching.com.tw.
dns        IN      A        192.168.1.197
www        IN      A        192.168.1.111
virtual    IN      CNAME    www

```

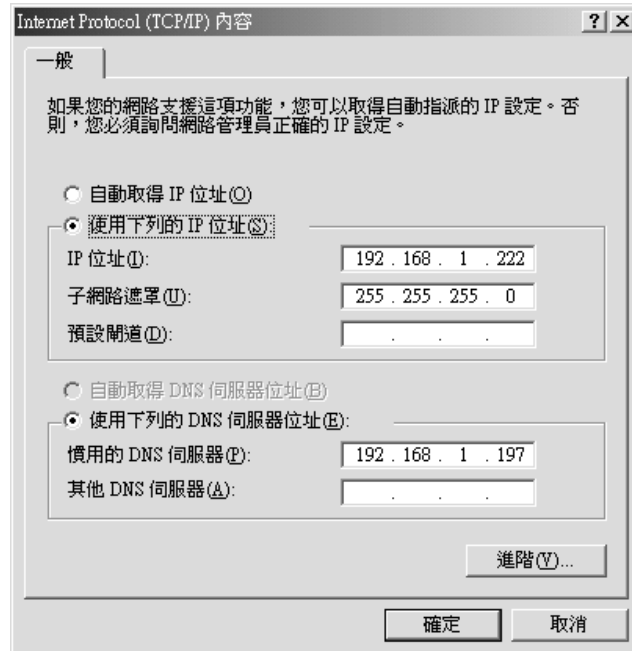
；關於 virtual 這台主機名稱也可以使用 A 紀錄。不過思考個問題，當虛擬主機有 5、6 台之多；時，而您都是使用 A 紀錄去設定的話，那將來主機位址異動時，就要修改好幾筆的資料，；相反的，使用 CNAME 時則只需修改一筆就行了。

(3) 啟動 named :

```
DNS:~ # rcnamed start
```

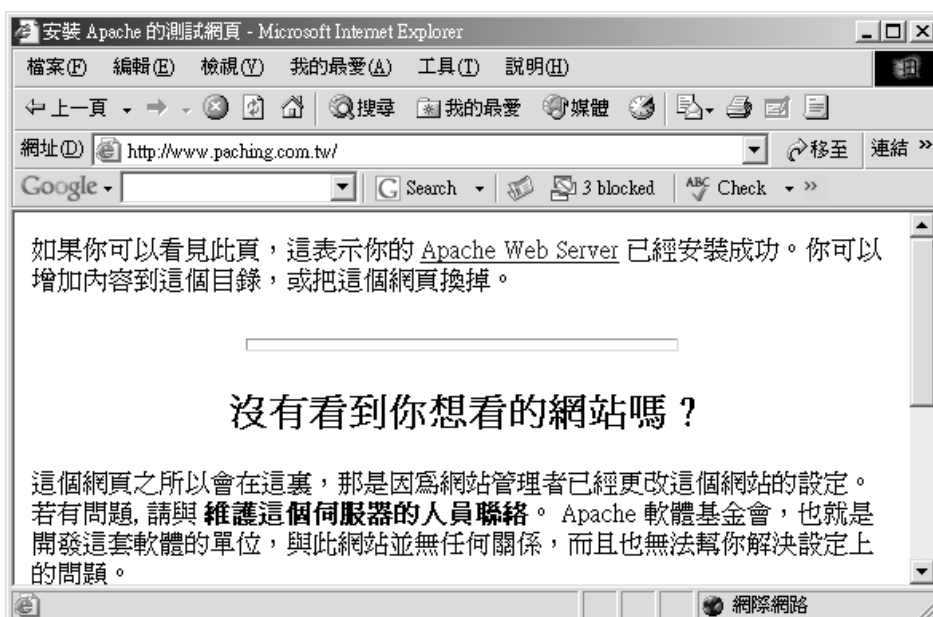
3. Client 端測試：

先指定好 Default Name Server：

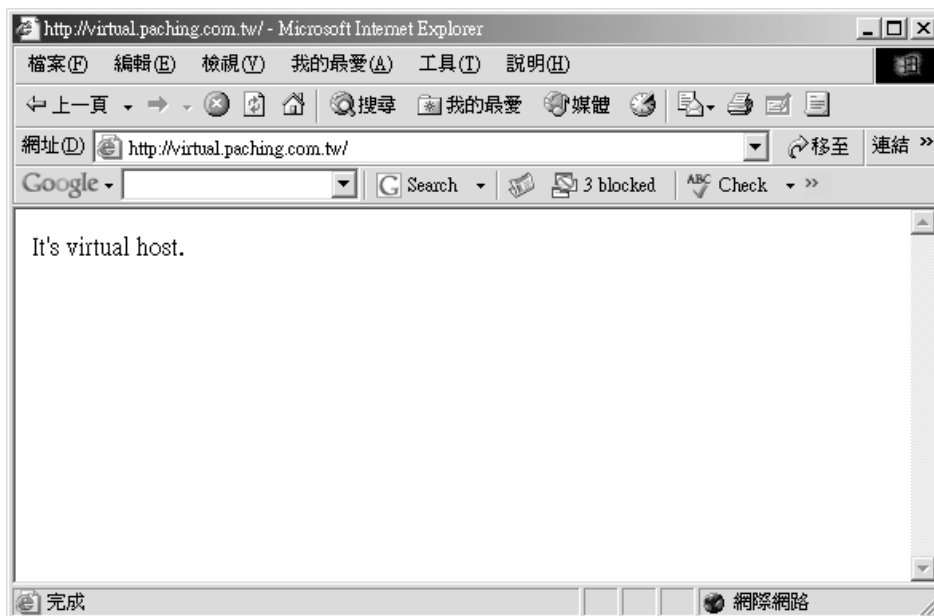


接著直接測試囉：

URL : <http://www.paching.com.tw/>



URL : <http://virtual.paching.com.tw/>



學會了虛擬主機的設定之後，是不是覺得很簡單呢，您有實做出來嗎？沒有的話，可要加把勁了。