

第九章

代理伺服器

這裡有幾個充分的理由你不該直接將忙碌的網站連上 Web：

- 將經常拜訪的網頁存入快取區，並將其餘的請求分散於許多伺服器，以得到較佳的效能。
- 給予壞人另一層障礙以增進安全性。
- 提供被防火牆保護的本地使用者存取 Web 的功能，如第十一章所討論的。

解決辦法是使用代理伺服器，它可以是 Apache 本身，或是類似 Squid 的特殊產品。

安全防護

Web 上的一項重要考量是不要讓壞人接觸你的網路（請參見第十一章）。一種已經成熟的技術是將網路藏在防火牆的後面；這可以順利地運作，不過，當你如此作時，也代表在同一網路的使用者，會突然發現他們無法連上 Web 了（很像住在邁阿密的人們，在高樓大廈紛紛興起前，可以瀏覽美麗的海灘風景，不過，後來就什麼都看不到了）。在 Butterflies 公司，隨著競爭的加劇，加上搗蛋的壞人一直想要突破安全防線，以入侵我們的系統，這已成為迫切的問題。我們安裝了防火牆，並可預見馬上就會得到業務部門的抱怨，因此我們也安裝了代理伺服器讓他們能連上 Web。

所以，除了 Apache 服務拜訪我們網站的客戶，並受防火牆保護外，我們還需要另外一部 Apache 伺服器當作代理伺服器，讓我們存取 Web 上的其它網站。如果沒有代理伺服器，雖然在防火牆內的人們是安全的，他們卻是像瞎子一樣什麼都看不到。

代理伺服器的指令

此處我們不關心防火牆，所以會視其為理所當然。重點是如何設定 Apache 的代理伺服器，讓防火牆後面的人們有好日子過。

`site.proxy` 有三個子目錄：`cache`、`proxy`，與 `real`。.../`site.proxy/proxy` 中 `Config` 檔的內容如下：

```
User webuser
Group webgroup
ServerName www.butterthlies.com

Port 8000
ProxyRequests on
CacheRoot /usr/www/APACHE3/site.proxy/cache
CacheSize 1000
```

值得注意的是：

- 在這個網站上，我們使用 `ServerName www.butterthlies.com`。
- 埠號設為 8000，所以不會與在同一機器上執行的真實 Web 伺服器衝突。
- 我們啟用 `ProxyRequests`，並替快取區設定一個目錄，這點在稍後會談到。
- `CacheRoot` 是在某一特別的目錄中設定。
- `CacheSize` 設為 1000 KB。

AllowCONNECT

```
AllowCONNECT port [port] ...
AllowCONNECT 443 563
伺服器組態、虛擬主機
相容性：AllowCONNECT 只適用於 Apache 1.3.2 及之後的版本。
```

`AllowCONNECT` 指令設定一串讓代理伺服器的 `CONNECT` 方法可以連接的埠號。當客戶端請求 `https` 連線，以及在 `http` 上建立出代理通道時，現今的瀏覽器都使用此法。

在預設情況下，只會啟用預設的 `https` 埠（443）及預設的 `snews` 埠（563）。使用 `AllowCONNECT` 指令可以改寫這些預設值，並只允許連線至所列出的埠。

ProxyRequests

```
ProxyRequests [on|off]
預設值：off
伺服器組態
```

本指令會啟用代理服務的功能。即使當 `ProxyRequests` 設為 `off` 時，仍會啟用 `ProxyPass` 指令。

ProxyRemote

```
ProxyRemote match remote-server
伺服器組態
```

本指令定義代理伺服器的遠端代理主機（亦即，應該用於某些請求，而非直接符合的代理伺服器）。`match` 可以是遠端伺服器支援的 URL 方法名稱、應該用於遠端伺服器的部分 URL，或是 `*` 以表示對於所有的請求都應該聯繫遠端伺服器。`remote-server` 是應該用來聯繫遠端伺服器的 URL（亦即，它是 `protocol://hostname[:port]` 的格式）。目前，只有 HTTP 可以作為遠端伺服器的通訊協定。例如：

```
ProxyRemote ftp http://ftpproxy.mydomain.com:8080
ProxyRemote http://goodguys.com/ http://mirrorguys.com:8000
ProxyRemote * http://cleversite.com
```

ProxyPass

```
ProxyPass path url
伺服器組態
```

本指令在一般的伺服器上執行，並將對某一目錄及其下之檔案的請求，轉換為對代理伺服器的請求。因此，在一般的 Butterthlies 網站上，我們可能會想要將對 `/secrets` 的請求傳給代理伺服器 `darkstar.com`：

```
ProxyPass /secrets http://darkstar.com
```

不幸的是，這沒有想像中那麼有用，因為代理伺服器不會修改由 `darkstar.com` 傳回的 HTML。這代表嵌入在 HTML 中的 URL 會指向主伺服器上的文件，除非它們被仔細編寫過。例如，假設 `one.html` 文件係用 `http://darkstar.com/one.html` 的 URL 儲存在 `darkstar.com` 上，而我們想要它能指向在同一目錄中的另一份文件。則當以 `http://www.butterthlies.com/secrets/one.html` 來存取網頁時，下列的連結可以成功；`one.html` 的內容為：

```
<A HREF="two.html">Two</A>
<A HREF="/secrets/two.html">Two</A>
<A HREF="http://darkstar.com/two.html">Two</A>
```

但是，此範例則行不通：

```
<A HREF="/two.html">Not two</A>
```

當經由 `http://darkstar.ccm/one.html` 直接存取網頁時，下列的連結可以成功：

```
<A HREF="two.html">Two</A>
<A HREF="/two.html">Two</A>
<A HREF="http://darkstar.com/two.html">Two</A>
```

不過，這樣則不行：

```
<A HREF="/secrets/two.html">Two</A>
```

ProxyDomain

```
ProxyDomain domain
伺服器組態
```

本指令似乎只對企業網路中的 Apache 代理伺服器才有用。ProxyDomain 指令設定 Apache 代理伺服器所隸屬的預設網域。如果客戶端送出請求給不含完整合格之網域名稱 (FQDN) 的主機，則對在名稱後面加上所設定 domain 之同一主機，會產生重導回應。此處的重點是，在企業網路上的使用者經常只在瀏覽器上輸入網域名稱的第一部分，但是伺服器卻需要完整合格之網域名稱才能適當地運作。

NoProxy

```
NoProxy { domain | subnet | ip_addr | hostname }
伺服器組態
```

NoProxy 指令設定一串以空白分隔之子網路、IP 位址、主機、及 (或) 網域。對滿足任何設定值之主機所送出的請求，都會直接處理，而不再轉送給 ProxyRemote 所設定的代理伺服器。

ProxyPassReverse

```
ProxyPassReverse path url
伺服器組態、虛擬主機
```

反向代理是一種將伺服器偽裝成其它伺服器的技術 — 也許因為「真實」的伺服器是躲在防火牆後面，又或者是因為你想要用不同的機器來服務網站的各個部分，但又不想讓人發覺。這也可以分擔幾部伺服器之間的負載 — 前端伺服器只接受請求，並轉送給一部後端伺服器。非必要的模組 `mod_rewrite` 具有一些特殊配備以支援這種功能。本指令

讓 Apache 能調整 Location 回應標頭中的 URL。如果 ProxyPass (或 *mod_rewrite*) 曾經用來執行反向代理的工作，本指令會重新編寫從反向代理伺服器傳回的 Location 標頭，讓它看起來好像是來自其它地方 (當然，在一般情況下就是本伺服器)。

ProxyVia

```
ProxyVia on|off|full|block
預設值：ProxyVia off
伺服器組態、虛擬主機
```

本指令控制代理伺服器所使用的 **Via:** HTTP 標頭。它主要是用來控制代理請求在代理伺服器間的流向。請參閱 RFC2068 (HTTP 1.1) 以瞭解 **Via:** 標頭的意義。

- 如果設為預設的 **off**，則不會作特殊的處理。如果請求或回應中含有 **Via:** 標頭，則會照原樣傳送。
- 如果設為 **on**，則對於目前的主機，在所有的請求與回應中會加入 **Via:** 標頭列。
- 如果設為 **full**，則所有產生的 **Via:** 標頭列會額外地讓 Apache 伺服器在其版本訊息中顯示 **Via:** 的註解欄位。
- 如果設為 **block**，則每一代理請求會移除其全部的 **Via:** 標頭列，而且不會產生新的 **Via:** 標頭。

ProxyReceiveBufferSize

```
ProxyReceiveBufferSize bytes
預設值：無
伺服器組態、虛擬主機
```

ProxyReceiveBufferSize 指令為對外的 HTTP 及 FTP 連線設定網路緩衝區的大小，以增加網路輸出量。它必須大於 512 或設為 0 (表示使用系統的預設緩衝區大小)。

範例

```
ProxyReceiveBufferSize 2048
```

ProxyBlock

```
ProxyBlock *|word|host|domain [word|host|domain] ...
預設值：無
伺服器組態、虛擬主機
```

`ProxyBlock` 指令設定一串以空白分隔的字、主機，及（或）網域。代理伺服器會擋住名稱中含有相符的字、主機，或網域之網站，而這些網站係準備處理對 HTTP、HTTPS，及 FTP 文件之請求。在啟動時，代理伺服器的模組也會企圖決定所列項目的 IP 位址（這些項目也可能是主機名稱），並將它們存入快取區以進行比對測試。例如：

```
ProxyBlock joes-garage.com some-host.co.uk rocky.wotsamattau.edu
```

如果是指向 IP 位址，則 `rocky.wotsamattau.edu` 也會符合。

請注意 `wotsamattau` 也足以符合 `wotsamattau.edu`。

另外，也需注意：

```
ProxyBlock *
```

會擋住所有網站的連線。

外表上的錯誤

當伺服器被設成代理伺服器時，像這樣的請求訊息：

```
GET http://someone.else.com/ HTTP/1.0
```

會被接受，並交由適當的 Web 伺服器執行代理作業。在預設狀況下，Apache 不會執行代理的工作，不過，它可以看起來像要準備進行代理作業 — 像前面的請求會被接受，並以預設的組態來處理。Apache 會假設 `someone.else.com` 是在目前機器上的虛擬主機。大家有時會以為這是一個錯誤，不過事實上，這是正確的行為。請注意，所服務的網頁與任何在同一機器上真正未知的虛擬主機所服務的網頁是一樣的，因此這不會影響安全性。

效能

藉著將傳入的網頁存入快取區，你可以提昇代理伺服器的效能。因此，當下一次需要某一網頁時，可以立即提供出去，而不需浪費時間再連上 Web。對於傳出的網頁也一樣，尤其是立刻從執行 CGI script 與存取資料庫產生的網頁（記住這會產生舊的內容，因此並非一直是你想要的）。

內部快取

另外一個使用代理伺服器的理由是將來自 Web 的資料存入快取區，以節省十分擁擠的頻寬，並改善存取伺服器的時間。不過，請注意，實際上為了節省頻寬，卻經常得付出存取時間的代價。

如前所示，使用巧妙地插入 Config 檔中的 `CacheRoot` 指令，以及具有適當權限的快取目錄，可以讓我們看看這個運作情形。我們從建立 `.../site.proxy/cache` 目錄開始，然後 Apache 以類似 `.../site.proxy/cache/d/ofj/gfqbz@49rZiy6LOCw` 的目錄結構來改進它。

`gfqbZ@49rZiy6LOCw` 檔含有下列內容：

```
320994B6 32098D95 3209956C 00000000 0000001E
X-URL: http://192.168.124.1/message
HTTP/1.0 200 OK
Date: Thu, 08 Aug 1996 07:18:14 GMT
Server: Apache/1.1.1
Content-length: 30
Last-modified Thu, 08 Aug 1996 06:47:49 GMT
```

```
I am a web site far out there
```

下一次當有人想存取 `http://192.168.124.1/message` 時，代理伺服器不需要從 Web 上找了；它只需從快取區中尋找。

這裡還有許多內部控管用的指令，亦能協助快取作業。

CacheRoot

```
CacheRoot directory
預設值：無
伺服器組態、虛擬主機
```

本指令設定包含快取檔案的目錄；它必須能被 Apache 寫入。

CacheSize

```
CacheSize size_in_kilobytes
預設值：5
伺服器組態、虛擬主機
```

本指令設定快取區的大小（以 KB 為單位）。你可以設定一個較大的數字，以儲存較多的內容，不過系統資源回收機制會將它減少到低於設定值。

CacheGcInterval

```
CacheGcInterval hours
預設值：永不
伺服器組態、虛擬主機
```

本指令設定 Apache 隔多久（以小時為單位）會檢查快取區，並在資料量超過 `CacheSize` 的設定值時，執行資源回收的動作。

CacheMaxExpire

CacheMaxExpire *hours*
預設值：24
伺服器組態、虛擬主機

本指令設定存入快取區的文件可以保留的時間。即使文件的過期日被設為更以後，此限制仍會強制執行。

CacheLastModifiedFactor

CacheLastModifiedFactor *factor*
預設值：0.1
伺服器組態、虛擬主機

如果文件未設定過期日，則系統會以檔案上次修改時間乘上 *factor* 的結果當作預估值。CacheMaxExpire 指令具有優先權。

CacheDefaultExpire

CacheDefaultExpire *hours*
預設值：1
伺服器組態、虛擬主機

如果文件被不支援過期日的通訊協定取回，則使用此值。CacheMaxExpire 不會改寫它。

CacheDirLevels 與 CacheDirLength

CacheDirLevels *number*
預設值：3
CacheDirLength *number*
預設值：1
伺服器組態、虛擬主機

代理伺服器的模組以 URL 的雜湊值為檔名，儲存其快取內容。檔案名稱會分割成 CacheDirLevels 層目錄，而每一層則使用 CacheDirLength 個字元。當擷取檔案時，這可以提昇效率（因為在多數系統上，扁平的檔案結構是非常慢的）。所以，舉例來說：

```
CacheDirLevels 3  
CacheDirLength 2
```

會將雜湊值「abcdefghijk」轉換成 *ab/cdef/ghijk*。實際上，真正的雜湊值有 22 個字元，而每一字元為可能的 64 (2^6) 個字元中的一個，所以三個階層，每層一個字元，會得到 2^{18} 個目錄。你應該調整這個數字，以搭配預期的快取區檔案數 (2^{18} 大約是 25 萬，因此適合最多可容納幾百萬個檔案的快取區)。

CacheNegotiatedDocs

CacheNegotiatedDocs
預設值：無
伺服器組態、虛擬主機

如果本指令出現在 Config 檔中，它允許代理伺服器將內容協商過的文件存入快取區。這可以表示，在那些代理伺服器後面的客戶端，能擷取並非最符合其功能的文件，不過，卻會讓快取動作更有效率。

本指令只適用於來自 HTTP 1.0 瀏覽器的請求。HTTP 1.1 則對協商過的文件，提供更佳的快取控制，而本指令對 HTTP 1.1 請求的回應沒有影響。請注意，現在幾乎已沒有 HTTP 1.0 的瀏覽器了。

NoCache

NoCache [host/domain] [host/domain] ...

本指令設定一串以空格分隔的主機及（或）網域。來自這些主機及（或）網域的文件不會被存入快取區，例如像一些傳送即時股市行情的網站。

設定

代理伺服器的快取目錄必須十分仔細的以 webuser 的使用者，及 webgroup 的群組身份來設定，因為它會由 webuser 這個不起眼的人士所存取（請參見第二章）。

現在必須要讓瀏覽器知道你準備要使用代理伺服器來存取 Web。例如，在 Netscape 中，點選「編輯」→「個人功能設定」→「進階」→「代理伺服器 (Proxy)」，核選「手動設定 Proxy」，然後在 HTTP 的文字方塊中輸入我們的代理伺服器的 IP 位址。此伺服器與我們的 Netscape 係位於同一網段：192.168.123。而這個伺服器的 IP 位址為 192.168.123.4。

接著在「埠號」的文字方塊中輸入 8000。

對於微軟的 Internet Explorer，則點選「工具」→「網際網路選項」→「連線」標籤頁→「區域網路（LAN）設定」，核選「Proxy 伺服器」方塊，接著按下「進階」鈕，然後如前述的方式設定 HTTP 代理伺服器。這就是架設一台真實的代理伺服器所要作的事情。

如我們所做的，你可能想要在真正實作之前，先架設模擬環境來測試一下。不過，用桌上型電腦模擬一部代理伺服器可不容易，而且當我們模擬它時，有一些構成要素與目前為止示範所用的要素功用不同。最後，我們有下列四個構成要素：

- 在 Windows 95 機器上執行的 Netscape。在一般情況下，某一在 Web 上的人士會試著存取我們的銷售網站；現在，它則模擬一位企圖連線出去的 Butterthlies 公司的員工。
- 一部假想的防火牆。
- 在 FreeBSD 機器上執行，並作為 Butterthlies 網站的代理伺服器的一份 Apache（網站：`.../site.proxy/proxy`）。
- 也在 FreeBSD 上執行的另外一份 Apache（網站：`.../site.proxy/real`），它模擬另外一個網站「out there」，這是我們企圖存取的。我們必須假想由於 Web 無限制的浪費，所以會將我們與此網站隔離。

在 `.../site.proxy/proxy` 中的組態正如前所示。因為作為代理伺服器的機器與執行 `.../site.proxy/real` 的機器是分在 Web 的兩端，我們必須將此代理主機放在另一埠上，亦即，傳統的 8000。

在 `.../proxy/real` 中的組態檔之內容為：

```
User webuser
Group webgroup
ServerName www.faraway.com

Listen www.faraway.com:80
DocumentRoot /usr/www/APACHE3/site.proxy/real/htdocs
```

在這個網站上，我們使用比較簡明的 Listen 指令，並一起指定伺服器名稱與埠號。

在一般情況下，`www.faraway.com` 會是在 Web 上遠處的一個網站。不過，在我們的例子中，這是在同一台機器上的假想網站。

在 `.../site.proxy/real/htdocs` 中，有一個檔案含有下列的訊息：

```
I am a web site far, far out there.
```

同時，在 `/etc/hosts` 中，也有一列內容：

```
192.168.124.1 www.faraway.com
```

這是用來模擬這個遙遠的網站之 DNS 註冊資訊。注意這個網站是在不同的網段上 (192.168.124)，而我們常用的網段則是 192.168.123，因此當企圖在 LAN 上存取它時，還需要一些幫忙。

現在，在 FreeBSD 的機器上，`/usr/www/lan_setup` 檔的內容為：

```
ifconfig ep0 192.168.123.2
ifconfig ep0 192.168.123.3 alias netmask 0xFFFFFFFF
ifconfig ep0 192.168.124.1 alias
```

現在要作的動作是：切換至 `../site.proxy/real`，並用 `./go` 啟動伺服器，然後再切換至 `../site.proxy/proxy`，也用 `./go` 啟動伺服器。在你的瀏覽器上，存取 `http://192.168.124.1/`。你應該會看到下面的結果：

```
Index of /
. Parent Directory
. message
```

如果選擇 `message`，我們會看到：

```
I am a web site far out there
```

很好，可是我們在欺騙自己嗎？進到瀏覽器的代理伺服器設定項目中，並移除下列的 IP 位址，以停用 HTTP 代理伺服器：

```
192.168.123.2
```

然後，再存取 `http://192.168.124.1/`。這時，你應該會得到某種網路錯誤的訊息。

這裡發生什麼事呢？我們要求瀏覽器擷取 `http://192.168.124.1/`。因為它是位於 192.168.123 的網段，所以無法找到這個位址。因此它會使用 192.168.123.2 上，位於埠號 8000 的代理伺服器。它會將這個訊息送給代理伺服器【註】：

```
GET http://192.168.124.1/ HTTP/1.0
```

在 FreeBSD 的機器上執行，並傾聽埠號 8000 的那一份 Apache，則會接受此訊息。因為那份 Apache 要服務代理伺服器的請求，所以它會重新將該請求傳遞給我們一直認為是已經連結的目的地：192.168.123.1（因為它是在同一機器上，所以是可以連結的）：

```
GET / HTTP/1.0
```

在實際狀況下，事情反而比較簡單：你只需執行第二及第三步，而且可以不理會基本的理論。當你做完這些動作後，請記得在瀏覽器的設定中將 HTTP 代理伺服器的 IP 位址刪除。

●.....
註 從 URL 中的 `http:` 可以將此訊息辨識為代理伺服器的請求。

反向代理

本節說明當你需要使用虛擬主機時，如何將後端的 *mod_perl* 伺服器架設成代理伺服器。請參閱 perl.apache.org/guide/scenario.html，我們已經任意地引用其中的文件。雖然最好在一開始就設定正確（亦即，對不同的伺服器使用不同的 URL），不過，至少有三點理由你可能會想要重新編寫：

1. 因為你第一次沒有想清楚，所以現在要補救。
2. 因為你想要藉著使用相對而非完整的 URL，以節省網頁的大小。
3. 你想要改進效能，例如，將執行費時的 CGI 之結果存入快取區。

「虛擬主機」一詞指的是在同一台機器上維持一個以上的伺服器的技術，而這些伺服器係以其主機名稱來區分。例如，共用某一 Web 伺服器的公司行號經常會想要有自己的網域，以 *www.company1.com* 及 *www.company2.com* 的方式存取 Web 伺服器，而不用要求使用者記住任何額外的路徑資訊。

一種作法是在後端伺服器上，對每一虛擬主機只使用唯一的埠號，因此你可以從前端伺服器重新導向至 *localhost:1234*，以及前端伺服器上以名稱定址的虛擬主機；不過，在前端伺服器上的任何技術都可以使用。

如果在同一機器上同時執行前端與後端伺服器，當你緊密地結合至 *127.0.0.1* (*localhost*) 時，可以防止任何從外部至後端伺服器的直接連線，如在下列的組態範例中所示的。

以下是前端（輕巧的）伺服器的組態：

```
<VirtualHost 10.10.10.10>
  ServerName www.example.com
  ServerAlias example.com
  RewriteEngine On
  RewriteOptions 'inherit'
  RewriteRule \.(gif|jpg|png|txt|html)$ - [last]
  RewriteRule ^/(.*)$ http://localhost:4077/$1 [proxy]
</VirtualHost>
<VirtualHost 10.10.10.10>
  ServerName foo.example.com
  RewriteEngine On
  RewriteOptions 'inherit'
  RewriteRule \.(gif|jpg|png|txt|html)$ - [last]
  RewriteRule ^/(.*)$ http://localhost:4078/$1 [proxy]
</VirtualHost>
```

這個前端組態處理兩個虛擬主機：*www.example.com* 及 *foo.example.com*。這兩個設定幾乎完全一樣。

前端伺服器會在內部處理副檔名為 *.gif*、*.jpg*、*.png*、*.txt*，及 *.html* 的檔案；其餘的檔案則交由後端伺服器以代理的方式來處理。

這兩個虛擬主機的設定之間唯一的差異是，前者會重寫對後端機器埠號 4077 的請求。而後者則針對埠號 4078。

如果你的伺服器是設定要執行傳統的 CGI script（在 *mod_cgi* 下），及 *mod_perl* 的 CGI 程式，則設定前端伺服器來直接執行傳統的 CGI script 會有好處。如果你全部的 *mod_cgi* script 都以 *.cgi* 作為副檔名，則可以改變 *gif|jpg|png|txt* 的 Rewrite 規則以在尾端加入 *|cgi*，或者加入新的規則以在本機處理所有 */cgi-bin/** 的位置。

這裡是後端（沈重的）伺服器的組態：

```
Port 80

PerlPostReadRequestHandler My::ProxyRemoteAddr

Listen 4077
<VirtualHost localhost:4077>
    ServerName www.example.com
    DocumentRoot /home/httpd/docs/www.example.com
    DirectoryIndex index.shtml index.html
</VirtualHost>

Listen 4078
<VirtualHost localhost:4078>
    ServerName foo.example.com
    DocumentRoot /home/httpd/docs/foo.example.com
    DirectoryIndex index.shtml index.html
</VirtualHost>
```

後端伺服器知道如何區分請求是針對哪一個虛擬主機，它會檢查傳送至代理伺服器的請求所攜帶的埠號，並使用適當的虛擬主機區段來處理它。

我們設定 Port 80，所以任何重導會以 80 作為 URL 的埠號，而非在後端伺服器上真正執行的埠號。

為了從代理伺服器取得真正的遠端 IP 位址，你必須依照 Apache 模組 *mod_proxy_add_forward* 來使用 *My::ProxyRemoteAddr* 處理器。在 *mod_perl* 1.22 之前，這個功能必須已經設定給虛擬主機，因為它並不會被虛擬主機所繼承。

下列的組態是另一個有用的變通範例。它指定欲送至代理伺服器的資訊，而其餘的則由前端伺服器提供服務：

```
RewriteEngine    on
RewriteLogLevel  0
RewriteRule      ^/(perl.*)$ http://127.0.0.1:8052/$1  [P,L]
NoCache          *
ProxyPassReverse / http://www.example.com/
```

因此，如在前例中所做的，我們不需要設定規則給由前端伺服器服務的靜態物件，以在內部處理副檔名為 *.gif*、*.jpg*、*.png*，及 *.txt* 的檔案。