網頁交換器產品評比 — 功能與效能面

林柏青 蔡品再 林盈達
國立交通大學資訊科學所
新竹市大學路 1001 號，郵遞區號 300
(03) 571-2121 轉 56667
{pachinko, motse, ydlin}@cis.nctu.edu.tw
主要聯絡人：林柏青

摘要

隨著上網人數不斷的激增，寬頻網路的普及，許多大型的內容提供者(ICP)所面臨的最大挑戰就是巨量的使用者需求。單一伺服器已不能滿足這樣龐大的需求，所以最好的解決方案就是伺服器叢集系統。L4 的伺服器叢集技術已發展了許多年，不過對於今日電子商務及其它的網路服務，L4 負載平衡技術不能根據使用者要求的內容做交換是不夠用的，甚至會導致嚴重的錯誤。爲了滿足現今的需求 L7 的伺服器叢集技術因應而生。本文邀請了六家商業產品做功能和效能上的比較，參與的廠商有 Nortel、Cisco、F5、Foundry、Radware 及 3Com。我們利用 WebBench 及 Sniffer 等工具對產品進行測試及分析比較。藉由這樣的測試，除了可以提供服務廠商在選購產品，以及設備廠商日後設計產品和訂定規格時做爲參考外，也可以協助學術界在相關技術找尋新的研究方向。

關鍵字：伺服器、交換器、負載平衡、URL、cookie、SSL、評比

一、需求與相關技術

1.1 網頁交換器的需求

上網人口的激增是目前各國有目共睹的事實，相對的使用者的需求也隨之增高，對於這麼大量的使用者需求，我們要怎麼去滿足它呢？伺服器叢集就是爲了解決這個問題所發展出來的技術，如 DNS round robin，NAT，direct routing 等技術[1]。這些技術可以根據連線的目的地 IP 位址和 TCP 埠號來做交換，當負載平衡器取得使用者要求的第一個 TCP syn 封包時就會依設定好的排程方式選擇後方一台適合的主機，將封包轉送到後面的伺服器，並且它會記住這個 socket pair 必需要重導到那台伺服器，以便後來同一個 TCP 連線的要求會送到同一台伺服器。不過這些技術都屬於 L3、L4 的負載平衡技術，可能會因爲看不到使用者需求的內容而導致交換上的錯誤。所以發展出 Layer7 的網頁交換器，它解決了 L4 負載平衡技術所造成的問題，並且能夠更聰明的做交換的動作。關於 L4 的問題，L7 的好處分述如下：

Cookie session persistence

現在的許多網頁都是由一些動態網頁程式(ASP、JSP、PHP 等)所寫成的，其中都會支援所謂

的 session 的功能，這個功能可以視做是 Server 端的 cookie，程式寫作人員可以把一些重要的資料如使用者資料、權限等存於 Server 端，這樣可以避免使用一般 cookie 造成的資料容易被取得的安全性問題。這個 session 是由 cookie 實作的，當使用者連線時會給使用者一個 session id 的 cookie，使用者之後的 request 都會帶著這個 cookie，Server 則是根據這個 id 來索引使用者的 session。所以對於這樣的應用，負載平衡器必需知道那個 server 產生了什麼樣的 cookie，在下次使用者的 request 來時，必需要根據 request 裡的 cookie 來做 switching，重導到原來的 server。而 L4 叢集技術因爲看不到 request 中的 cookie 所以可能會把它重導到另一台 server，這樣就造成了很嚴重的錯誤，因爲使用者的資料是存放在原來那台 server 裡的。這樣的情況常見於電子商務網站中，試想，當使用者要結帳時發現他精心挑選的產品不見了，是何感想？

內容切割(content partition)與快取交換(cache switching)

隨著多媒體與寬頻網路的快速發展，許多網站的內容也逐漸肥大，爲了有效儲存這些內容，就有內容切割的想法。我們可以根據 URL 得知使用者要求的內容，然後把要求正確地導向儲存該內容的伺服器，如我們可以把 content 分爲 html、cgi，分別存於二個不同的 server group，如圖一，而不用把整個網站的內容存於同一台 server。根據[2]所提到的 Locality aware 就是類似這樣的架構，這樣的架構可以提高伺服器的效能。要支援這樣的功能，switch 必需要根據 request 的 URL 做 switching 的動作，這也是 L4 叢集技術所無法做到的事。根據 URL 做 switching 還有另一個重要的應用，就是快取交換，大型 ISP 爲了加速網路的回應都會架設許多的 forward proxy cache server，如果使用 URL switching 的技術，可以設定每一台 cache server 負責那些 URL 的 request，這樣每台 server 所 cache 的內容就會完全不一樣，可以達到很高的效能。此外也可以直接 bypass 不能 cache 的內容到 Internet 裡。

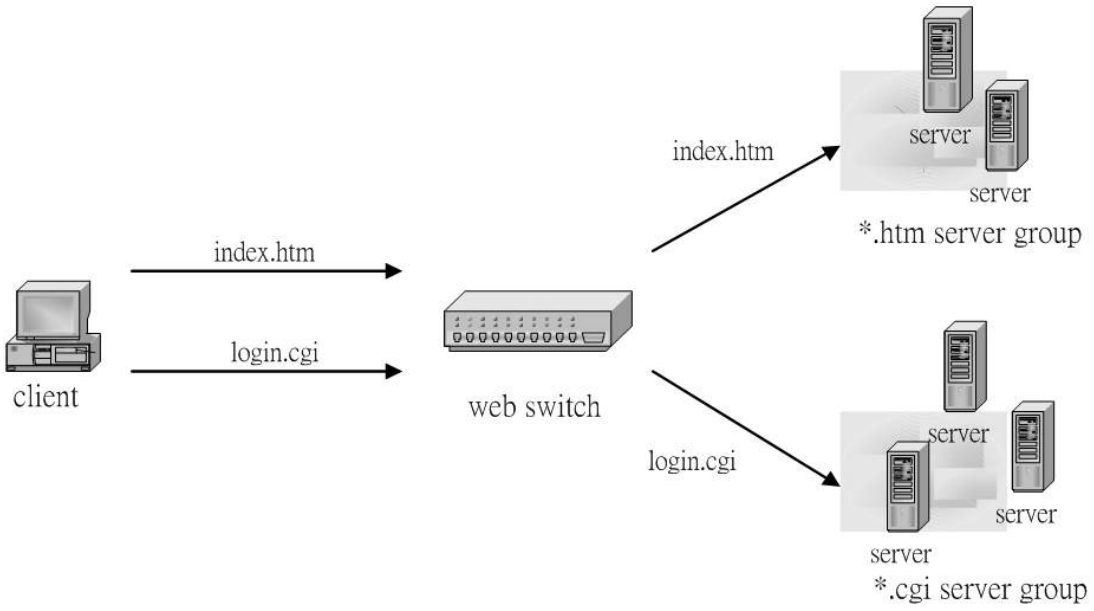


圖 1. 利用 URL 做負載平衡的應用

SSL 交換

在電子商務中，爲了確保交易時信用卡資料不被盜取，常常會使用 SSL 技術做爲傳送資料的協定，不過 SSL 演算法必需要花掉許多的計算的時間。爲了減少時間，SSL 提供重覆使用同一個 session 的功能[3]。爲了能夠重覆使用同一個 session，switch 必需要記錄那一個 SSL session id 要導向那一台 server(原來保有 session 的那台)。這也必需要網頁交換器的支援，因爲 L4 叢集技術是無法根據 SSL session id 作 switching 的。

支援 HTTP 持續連線(persistence HTTP)

HTTP 協定爲了提高效能，提供了所謂的 Keep-Alive connection [4]，讓一個 TCP connection 能夠傳送許多個要求。這對一般的 L4 叢集技術來說並沒有什麼多大的問題，不過對於網頁交換器卻是一個很大的問題。例如在同一個 connection 原本要求 a.htm，所以一開始就將 traffic 導向負責 html 的 server 裡，接下來的 request 如果要求 b.gif 就必需要把原來導向 html server 的 connection 再重導到負責 gif 的 server[5]。這個問題似乎沒有很固定的解法，不過在[5]之中的解法是最爛的。因爲如果後方的 server 全部使用了 NFS 將所有的 file system 連接在一起的話，那就不用做 content partition 了。在這次的測試中也沒有任何一家採用[5]的方式。

1.2 L4 與 L7 負載平衡技術比較

延遲連接(Delay binding)

實做網頁交換器最難的部分是，使用者的 request 內容是在 TCP 建立完成後才會送出來的，也就是因爲這樣，所以不然像 L4 一樣，可以直接把 TCP 建立的動作 forward 給後方的 server。所以網頁交換器必需要和用戶端建立完 TCP 連線後，取得使用者 request 內容，在依此內容選擇後方 server，此時由網頁交換器跟後方的 server 建立另一個 TCP 連線，再將用戶端 request 的封包的 sequence number、destination IP 改成網頁交換器與後方 server 建立的 TCP 連線的 sequence number、destination IP，然後送給後方 server，如圖二。這個方法叫做 delay binding[6]。在此我們比較 L4 與 L7 對封包處理的流程於表一。

在表一中，斜體字爲網頁交換器在重導第一個封包到後面伺服器前要做的事。其中 L4 和 L7 都會做的事，如伺服器的選擇，建立 connection table，修改封包等是一樣，但在 L7 還要負責與後方的 server 建立 TCP 連線，要回應用戶端 syn-ack 等是 L4 所不需要做的事。對於同一個 TCP 連線上後續的使用者要求，L7 還要對 parse 每個要求是不是存在目前的伺服器上，否則又得要再重導向一次。因 parse 要求需做字串的搜尋比對，這動作遠比 L4 固定欄位的比對來的複雜。而 L4 只需要查 connection table 並把要求轉送個目前的伺服器就可以了。在 L7 所要做的處理遠多於 L4。

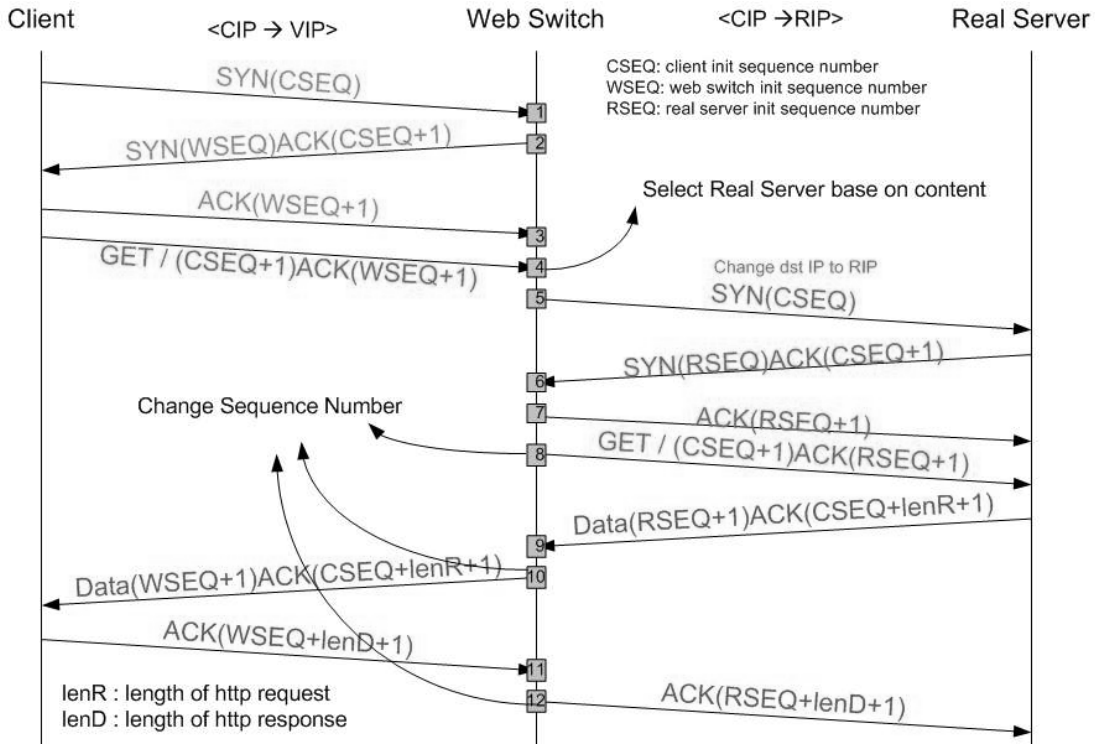


圖 2. Delay Binding 流程

	L4	L7
Client → syn	*1. 選擇一台伺服器* *2. 建立 connection table* *3. 修改 dst.IP，forward 封包至 server* 4. 收到伺服器的 syn-ack，修改 src.IP，forward 封包回 client	*1.回傳 syn-ack 封包，並等待 client 端的 ack 封包以完成 3-way handshake。*
Client → get / http1.1	1. 檢查是不是存在的 connection，否則就 drop it 2. 查 connection table 找到同一台伺服器，forward 封包	*1. parse URL 並根據 URL 選擇一台伺服器* *2. 建立 connection table* *3. 向伺服器建立 TCP 連線* *4. 修改 det.IP、seq no.、tcp checksum，forward 封包*
Client → get /logo.gif http1.1	1. 檢查是不是存在的 connection，否則就 drop it 2. 查 connection table 找到同一台伺服器，forward 封包	1. 檢查 content 內容，如果並不存在現在連線的伺服器中則再重導向到新的伺服器，否則重導到原來的伺服器。

表 1. L4 與 L7 交換器處理封包方式比較表

二、測試對象

在篩選產品的過程當中，我們首先上網搜尋相關產品網頁，找出能依據 HTTP 要求的內容做爲選擇伺服器(即 content aware)的產品，約有十來家，相關產品的網頁可從[7]當中找到。其中台灣有代理的共七家，但因 Intel NetStructure 並沒有聯絡到合適的代理商，最後只邀請到其餘六家。選定產品後，由網路通訊雜誌社出面對各廠商發出邀請，並附上我們的測試計畫書。邀請的廠商與產品名稱列於表二，各家產品的架構列於表三。

製造廠商	產品名稱	聯絡/代理廠商
Nortel	ACE switch 180e [8]	豪勉科技
Cisco	ArrowPoint CSS 11154 [9]	聚碩科技
F5	BIG-IP HA+ Controller [10]	F5 台灣分公司
Foundry	ServerIron XL [11]	懇懋科技
Radware	WSD Pro [12]	寅騰科技
3Com	SuperStack 3 Load Balancer [13]	3Com 台灣分公司

表 2. 參與廠商及產品一覽表

Company/ Product	Nortel ACE switch 180e	Cisco CSS 11154	F5 BIG-IP HA+ Controller	Foundry ServerIron XL	Radware WSD Pro	3Com SuperStack 3 load balancer
10/100M Port #	8	12	2	8	8	12
1G Port #	9*	2	2 (optional)	2	2	2
Processor	2 RISCs/port	Central RISC, Distributed forwarding	Pentium !!! 850MHz	Power PC MPC 740 400 MHz	Power PC MPC 750 266MHz	ASIC-based
OS/firmware	WebOS 9.0.38	WebNS	BSDI 4.0beta	7.03.02	Software version 7.10	A3.5
Memory	2MB per port	128 MB	512MB DRAM	32 MB DRAM	64MB DRAM	32 MB DRAM
Size/Weight	2U/8 kg	1.5U	2U/12 kg	1.5U/8 kg	1U/3.5 kg	1.5U/6.6 kg
Price	USD $29327 NT $1012K	NT $1250K	USD $24250 NT $837K	NT $374K	NT $1095K	USD $7995 NT$276K

*每個 port 只能從 10/100 Mbps 或 1 Gbps 兩種介面中擇一使用。

表 3. 產品架構比較表

這裡面較特別的是 F5 的 BIG-IP Controller，它是一個內部爲純 PC 架構的網頁交換器，藉由修改 BSD 核心及相關 daemon 來實做相關功能。此外，除了 F5 和 Cisco 兩家產品用硬碟儲存設定外，其餘的產品皆是把設定存在 flash 記憶體中，在開機的時候速度亦較快。

三、測試環境與方法

3.1 測試環境

圖三爲測試環境的照片，其中我們用多部 Booksize PC 做爲用戶端，另幾部做爲伺服器端，PC 的硬體規格如附錄一。用戶端及伺服器端皆安裝 Windows 2000 Server 版作業系統及，伺服器端另裝 IIS 5.0 提供網站服務。所用的工具軟體列舉如下：

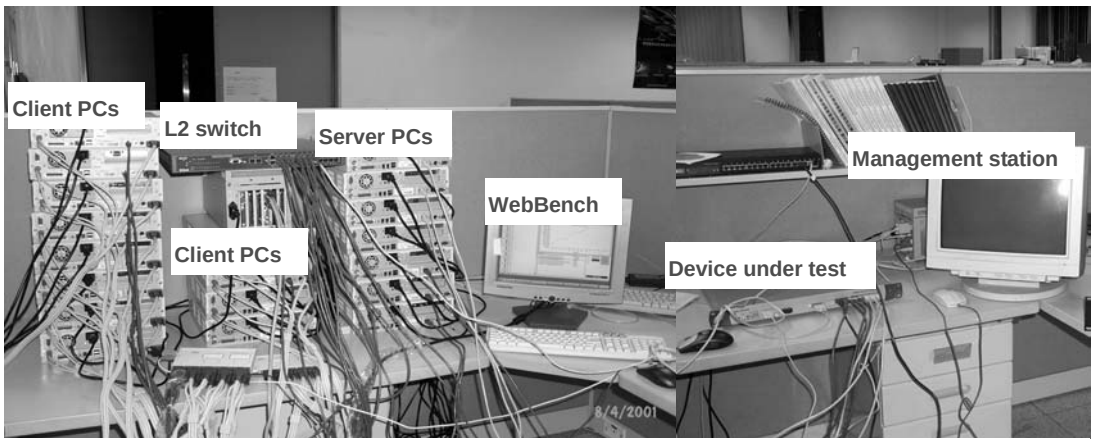


圖 3. 測試環境的照片

3.2 測試工具

1. WebBench：可快速且大量的產生 HTTP 要求(其產生的 workload 請參考附錄四)，並做記錄。

2.Sniffer Pro 4.5：用來監看網頁交換器處理封包的方式，及測量封包的延遲時間。

3.IE 瀏覽器 Version 5.5：用以配合 sniffer 產生不同的 traffic。

4.IIS 5.0：支援 HTTP 1.1 並把 log 關閉以達到最好的效能。

3.3 L4 負載平衡效能測試

目的：這項測試的目的，除了能了解各家產品在做 L4 負載平衡的表現外。更重要的，可對照在 L7 的處理下，對交換器效能的影響如何。我們分 NAT 及 direct routing 兩種方法進行測試。本次測試的產品當中，計有 Nortel、F5、Foundry、Radware 四家廠商的產品有提供 direct routing 的功能。此外，我們統一用 round robin 做爲分派伺服器的準則。

測試平台的設定：

(a) direct routing

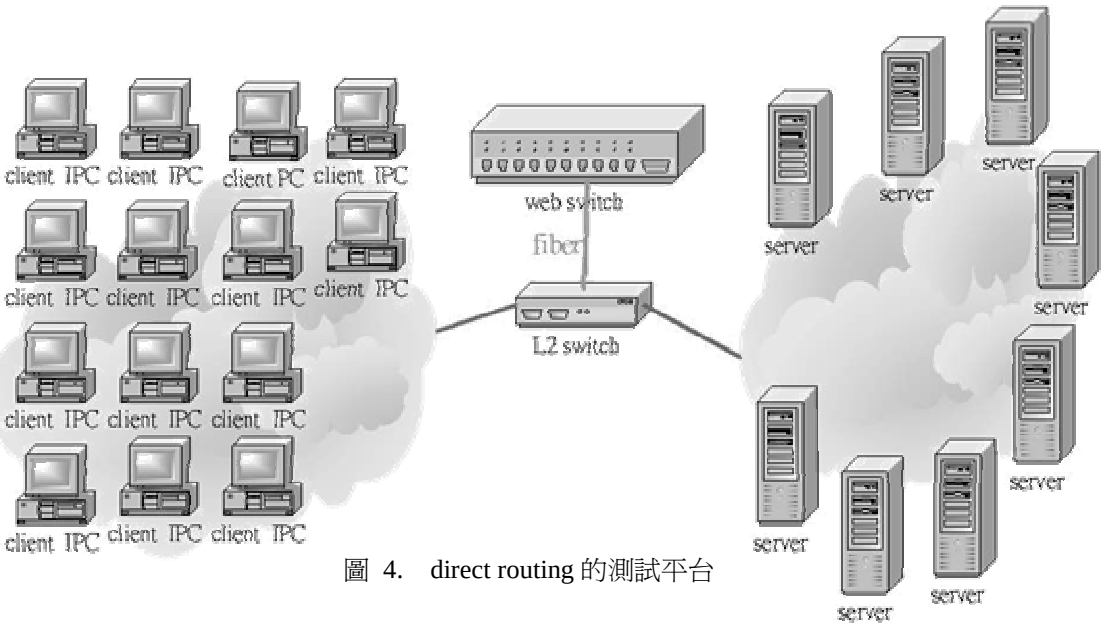


圖 4. direct routing 的測試平台

以 14 台 Booksize PC 做爲用戶端，另 8 台 Booksize PC 當伺服器，測試平台的設定如圖四。用戶端與伺服器皆直接接到 L2 交換器上，而 L2 拉一條 Gigabit Ethernet (1000BASE-SX) 的光纖到網頁交換器的 Gigabit port。HTTP 要求即進出該條光纖從用戶端經由網頁交換器到達伺服器，而伺服器的回應則經由 L2 交換器直接回到用戶端，無需經由網頁交換器。其中在設定上要特別注意的是伺服器端另外得設一個 loopback 位址和網頁交換器的虛擬 IP 位址相同，這是 direct routing 的基本要求。

(b) NAT load balancing

這項測試測試平台的設定如圖五，與圖三很不同的是伺服器端回傳的內容也會經過網頁交換器。此外，伺服器的 IP 位址跟虛擬 IP 位址屬於不同的 subnet。因爲在實際應用上，虛擬 IP 位址爲公開的 IP 位址，而伺服器通常使用私有的 IP 位址，兩者顯然屬於不同的 subnet，因此這樣的設定較符合真實狀況。

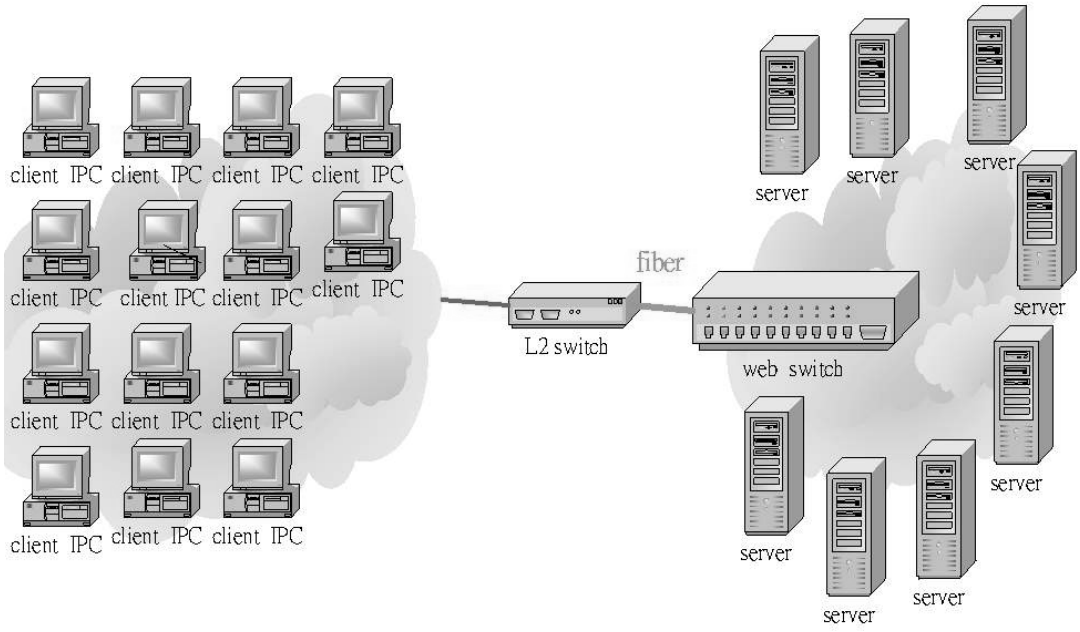


圖 5. NAT 負載平衡的測試平台

測試方法：由 14 台用戶端的 PC 同時用 WebBench client 產生大量的 HTTP 要求。因在實際測試時，發現 WebBench 在 HTTP 1.0 下因受限於每一個 HTTP 要求均需要做 TCP 連結及拆除的動作，使得用戶端無法同時產生太多的 HTTP 要求。因此這項測試中，我們使用 HTTP 1.1 的持續連結的方式來產生更大量的封包進行比較。對每項測試，我們都至少進行若干遍，並將所得的結果取其平均。

3.4 L7 負載平衡效能測試

目的：主要是測試各家產品在處理封包中 URL 的訊息時所表現的效能，並跟 L4 負載平衡的情況做比較。我們也同時做 HTTP 1.0 的測試，並與 HTTP 1.1 的結果對照，以突顯各家產

品在 HTTP 1.1 的情況下，對效能的改進程度。並且也測試了不同 URL 規則的個數下的效能，以期了解各家產品是否會因 URL 個數的增多而影響效能。3Com 的產品因爲不支援 URL 交換的功能，所以不列入本次測試。

測試平台的設定：測試平台的 IP 位址設定與拉線的方式與圖五類似，但伺服器減爲六台。因爲實測結果發現六台已足夠處理進來的要求。因各家產品 URL 規則設定方式並不完全相同，我們歸納後發現用副檔名做規則是各項產品皆共有的，也符合實際應用的需求。因此我們統一以底下兩種狀況的需求做爲測試的標準，並以 round robin 做爲分派方式。

I. 若 HTTP 要求是以.gif 做副檔名，則分派到三台伺服器中的一台。若是以.htm 做副檔名，則分派到另外三台伺服器中的一台。其餘的要求皆分派到六台伺服器中的一台。
II. 對.gif 與.htm 要求的處理規則同上。但新增若以.aaa 做副檔名，則分派到六台伺服器中的一台；若以.aab 做副檔名，則分派到六台中的一台……依此類推，直到有 50 條 URL 規則爲止。

由於 WebBench 產生的要求只有.gif 和.htm 兩種，並無如.aaa 等奇特的副檔名，所以語意上這兩種分派的結果應該是相同的。藉由這樣的設定，可將唯一的差異控制在只有規則的個數上。希望能藉此了解規則個數對各家產品效能的影響有多大。其中測試狀況 II 的目的較學術性，主要是希望了解規則個數對效能的影響，在真實應用情況不大可能發生。

測試方法：同樣是由 14 台用戶端的 PC 同時用 WebBench client 產生大量的 HTTP 要求，我們針對 HTTP 1.0、1.1，以及前段描述的狀況一、二，共四種情形進行測試。

3.5 cookie 封包處理效能測試

目的：了解各家產品對於類似 ASP session 問題的支援如何，效能又是多好。

測試平台的設定：網路架構如圖五，14 台 client，6 台 server，不做 content partition。啓動各家的方法，使用 WebBench 4.1 來做測試。Workload 爲一個 6k 的 asp file，這個 asp 程式會向 client 送出 cookie。

測試方法：以 WebBench 4.1 產生大量的 workload，測試各家的效能，並用 sniffer 觀察各家的作法。

四、功能面比較結果

4.1 功能面比較

方法：在功能面方面，我們以表格條列的方式比較各家在 L4 的功能、L7 的功能、容錯、其他(如管理)等項的支援程度和差異，資料來源爲各家手冊。

結果：列於表四、五、六、七之中

Company/ Product	Nortel ACE switch 180e	Cisco CSS 11154	F5 BIG-IP HA+ Controller	Foundry ServerIron XL	Radware WSD Pro	3Com SuperStack 3 load balancer
NAT	Yes	Yes	Yes	Yes	Yes	Yes
Direct routing	Yes	No	Yes	Yes	Yes	No
Global server load balancing	Yes	Yes	Yes	Yes	Yes (for WSD-DS)	No
Dispatch metrics	min_miss, hash, least connections, round robin, quickest response bandwidth, weight	round-robin^{w}, least connections, hash	round-robin, least connections, quickest response predictive, weighr	round robin, least connections^{w}, quickest response^{w}	round-robin, least traffic, least users number, server load	round robin^{w} least connection^{w} quickest response
Firewall load balancing	Yes	Yes	Yes	Yes	No	No
VPN load balancing	Yes	N/A	Yes	No	No	No

表 4. L4 的功能比較表 (右上角的 w 表示可對該分配方式給各伺服器不同的 weight)

Company/ Product	Nortel ACE switch 180e	Cisco CSS 11154	F5 BIG-IP HA+ Controller	Foundry ServerIron XL	Radware WSD Pro	3Com SuperStack 3 load balancer
URL switching	Yes	Yes	Yes	Yes	Yes	No
Cookie persistence	Yes	Yes	Yes	Yes	Yes	Yes
SSL ID persistence	Yes	Yes	Yes	Yes	Yes	Yes
Matching rules	directory, substring, regular expression	directory, substring, file extension, regular expression	directory, substring, file extension	directory, substring, file extension	file extension, hostname	
Number of rules	128 strings, 40 bytes each rule, 4500 bytes each request	512 rules,1024 bytes each	no limit	256	256	
Rule precedence	Yes	Yes	No	Yes	No	

表 5. L7 的功能比較表

表中配對規則(matching rule)指的是規則的型態，如依照目錄、副檔名、目錄中的某個字串等。規則的優先權(rule precedence)則爲在設定時，是否可以排定規則的優先次序。

Company/ Product	Nortel ACE switch 180e	Cisco CSS 11154	F5 BIG-IP HA+ Controller	Foundry ServerIron XL	Radware WSD Pro	3Com SuperStack 3 load balancer
Active/active redundancy	Yes	Yes	Yes	Yes	Yes	Yes
Active/backup redundancy	Yes	Yes	Yes	Yes	Yes	Yes
Server check (ping)	Yes	Yes	Yes	Yes	Yes	Yes
Server check (port)	Yes	Yes	Yes	Yes	Yes	Yes
Server check (web page)	Yes	Yes	Yes	Yes	Yes	Yes
Server check (content)	Yes	Yes	Yes	Yes	Yes	No

表 6. 容錯的功能比較表

這裡的容錯包含網頁交換器本身的容錯和伺服器的容錯。網頁交換器的容錯包括可以用兩台以上交換器同時工作(active/active)，或是有的交換器在旁待命接替(active/backup)。伺服器的容錯包含下列四種

- ping : 網頁交換器發出 ping 的訊息，檢查是否伺服器有回應
- port: 網頁交換器試圖對某 port 建立 TCP 連結，看看是否能成功。
- web page： 網頁交換器試圖去取得一個 web page，看看是否能成功。
- content： 網頁交換器去分析取回來的 web page 內容，做爲成功與否的依據。

網管人員可以設定多久做一次伺服器的檢查，及連續檢查幾次失敗後宣告伺服器無法運作。

較難歸類的功能比較則整理於表七。

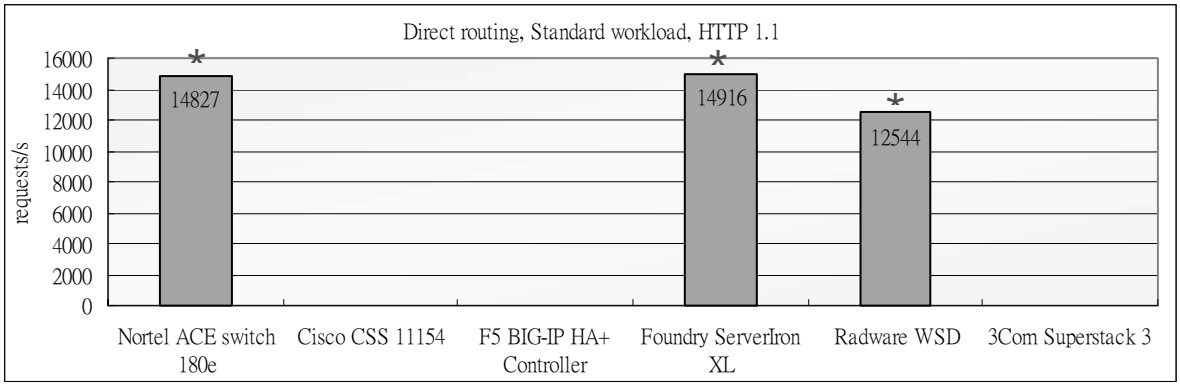
Company/ Product	Nortel ACE switch 180e	Cisco CSS 11154	F5 BIG-IP HA+ Controller	Foundry ServerIron XL	Radware WSD Pro	3Com SuperStack 3 load balancer
Bandwidth Management	Yes	N/A	Yes	No	Yes	No
IP Routing	Yes	Yes	Yes	Yes	Yes	Yes
Secure management	Yes	Yes	Yes	Yes	No	N/A
Packet filtering	Yes	Yes	Yes	Yes	Yes	Yes
Anti Dos	Yes	Yes	Yes	N/A	Yes	Yes
Configuration method	CLI, Java	CLI, Web	CLI, Web	CLI, Web	CLI, Web	CLI, Web

表 7. 其他的功能比較表

五、效能面測試結果

5.1 HTTP 1.1 下，direct routing 的效能

由於只有 Nortel, F5, Foundry, Radware 四家產品有提供 direct routing 的功能，所以我們只測試這四家。但其中 F5 因系統爲測試版在設定上有問題，故只有三家的結果。測試結果如圖六。

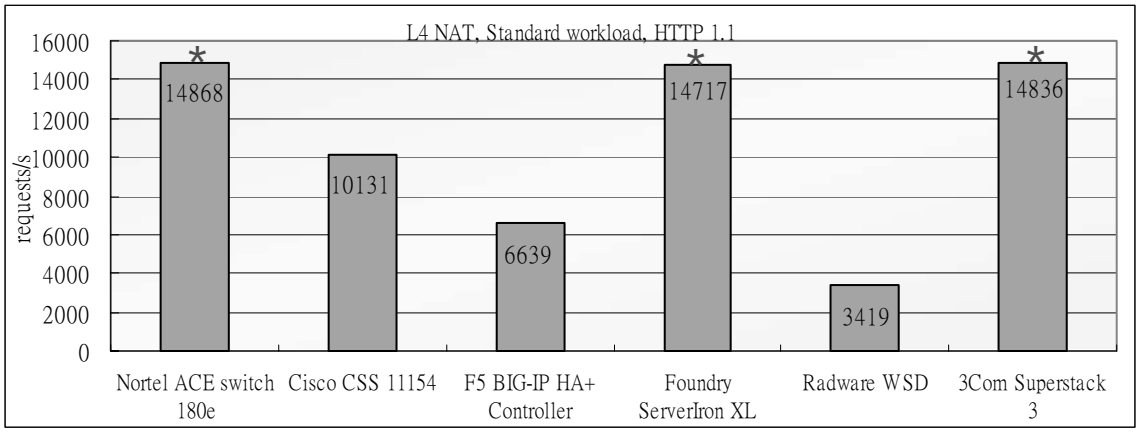


(*表示該數據受限於測試平台的能力，不是該產品真正的能力)

圖 6. direct routing 下，每秒可處理的要求個數

由於 L4 的封包處理很單純，所以受測產品的表現都超出我們的測試平台所能測試的範圍。從延遲時間的角度來看，Nortel 和 Foundry 的表現不相上下，在最大的 request rate 的平均延遲時間分別爲 5.218ms 及 5.180ms。Radware 的平均延遲時間略長，爲 5.501ms，使得用戶端 PC 能發出要求的速率略低於前兩家(因爲即使是持續連結，下一個 HTTP 要求也要等到上一個處理完後才能送出)，所以圖六的值會略低於前兩家。

5.2 L4 HTTP 1.1 下，NAT 負載平衡的效能



(*表示該數據受限於測試平台的能力，不是該產品真正的能力)

圖 7. NAT 負載平衡下，每秒可處理的要求個數

這裡面 Nortel、Foundry、3Com 三家產品表現仍然相當耀眼，甚至在平均延遲時間上接近 direct routing 的數據(三家分別爲 5.236ms、5.400ms、5.206ms)。Cisco 的值略低的原因據推測可能受限於 uplink port 上 IC 的處理能力，推測的原因是我們曾嘗試過把部份用戶端的 PC 直接接到 Cisco 上，發現 request rate 可以再略爲提高。F5 因爲是屬於純 PC 的架構，雖然我們另外借調 gigabit 的網路卡做測試，但效能仍未達到 gigabit 的速度，推測問題應該是受限在內部的 PCI 上。我們發現 Radware 在大量的 HTTP 要求湧入時，延遲時間快速的增長。在用 14 台用戶端 PC 送同時封包時，延遲時間高達 18.589ms。相對於它在 direct routing 相當好的表現，我們認爲它對大量封包的情況並不 scalable，因爲回應的資料量通常會遠大於要求的資料量。

5.3 L4 與 L7 的效能比較

結果如圖八所示，由於 3Com 的產品未支援 URL switching 的功能，故其 L7 數據的部份爲 0。因爲 URL switching 的處理較 L4 的處理複雜，所以在效能上都大幅度的下降。其中以 Foundry 的產品最爲嚴重，下降的幅度最高且在 L7 沒有很好的表現。對設備或晶片設計廠商而言，這意味著 URL 的處理方式或架構仍有值得開發的空間。對服務廠商而言，這意味著在使用 URL 做負載平衡時，除了設定上的彈性之外，也應當注意效能上是否能滿足需求。

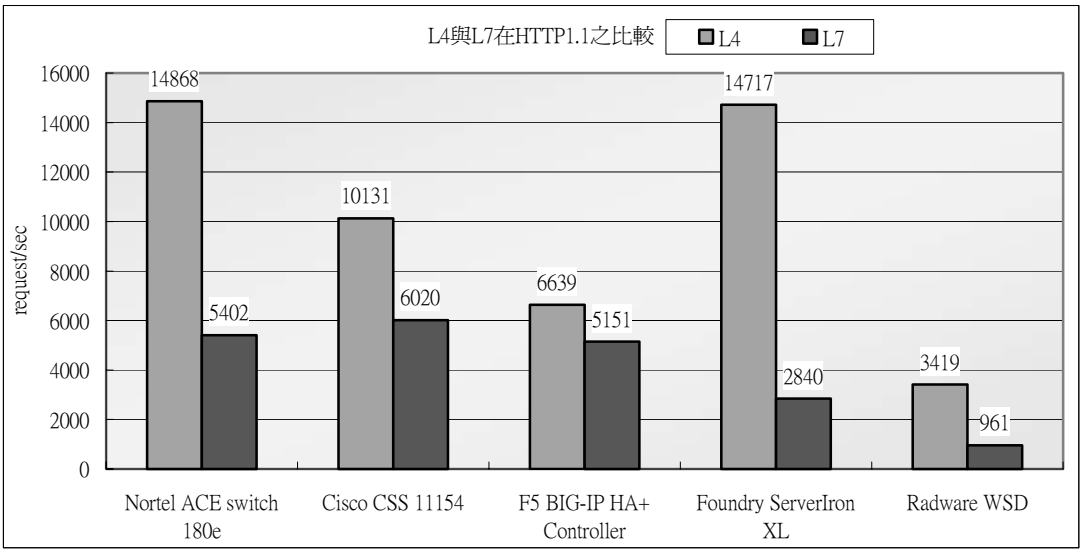


圖 8. L4 與 L7(3 rules)在 HTTP1.1 下之比較

5.4 各家產品相容 HTTP 1.1 的效能比較

如同前面「相關技術」中所提到的網頁交換器在 L7 Keep-Alive connection 必需要有特別的支援，在送測的六家產品各有不太相同的作法，不過可概分爲三類：

- 將 HTTP1.1 Keep-Alive connection 降級爲 HTTP1.0 connection close：這個做法很直覺，因爲要支援 Keep-Alive connection 要做特別的處理，所以就把它改成 HTTP1.0。不過這樣的做法不是很好，因爲 Keep-Alive connection 作法能夠減少延遲與頻寬的浪費，增加伺服器的效能，如果把它改爲不支援 Keep-Alive 的話，效能會很差。這樣作的產品有 Nortel、Foundry 和 Radware，第一家是在每一個 request 後由網頁交換器發 TCP RST 給伺服器來結束 connection，後二家是直接改 HTTP Header，將 connection: Keep-Alive 改爲 connection: close。第一種作法的效果會比第二種好一點，因爲從用戶端到網頁交換器之間還是 Keep-Alive connection。

- 發 HTTP 302 重導訊息：這個作法是當網頁交換器發現使用者所要求的內容不在目前連結的這個伺服器上時，就會向用戶端發出 HTTP 302 Object moved，要求用戶端把 connection 重導向到另外一個 URL。這個作法比前一個好很多，因爲只要在要求的 content 還在同一台伺服器的話，它還是保留 Keep-Alive connection 的特性。Cisco 就是屬於這種方法。

- 記住每一台伺服器的狀態：網頁交換器會記住所有曾經建過連線的伺服器的 connection 狀態，如果還會再重導到同一台的話，它會使用舊的狀態，接續舊的 connection。這樣作法以完全保留 Keep-Alive connection 的特性，F5 就是屬於這樣的作法。

圖九是 L7 HTTP1.0 與 1.1 的效能比較，Nortel 約增加了 16%的效能，Cisco 成長了 100%!!。這可以看可第二種作法比第一種作法好上很多。使用第三種作法的 F5 成長 10 倍？不過這個數據

不能參考，因爲 F5 的產品在 HTTP 1.0 的效能不佳，據用 Sniffer 觀測的結果發現在 HTTP 1.0 下，BIG-IP 在收到伺服器的回應之後，仍會重覆發送同樣的 HTTP 要求給伺服器，造成諸多不必要的封包。據 F5 台灣分公司回應表示，所借測的系統版本爲新版的測試版，因此我們推測這應該是一個測試版中的 bug，也許也是因爲這個原因，所以在測試過程中發生了許多次的 Kernel panic 的當機事件。

在測試過程中，Cisco 更新過系統版本，原來送測的是約 1 年半前的系統(ver 3.10)，在此項測試中效能只增加了 36%，新版的系統(ver 5.10)效能比較好。不過我們發現了一個系統的 bug，就是當網頁交換器發現所要求的 content 不在連線的伺服器時，必需要發出 HTTP 302 重導的訊息，並把目前和伺服器的連線切斷，Cisco 會發出 TCP RST 的封包給伺服器，不過 Cisco 的這個封包裡的 Sequence number 並不正確，使得後方的伺服器不接受這個封包，也就沒有把連線給斷掉。這造成了伺服器 TCP connection timeout，並發生重傳的現象。這並不會造成網頁交換器的 workload 不過對伺服器卻是一大傷害。會發生這種現象另一個原因是 TCP 的 ack piggyback 在 request 之中，而這個 request 所要求的內容又在別台伺服器，如果網頁交換器注意到這個問題，並自己發 ack 給原來的伺服器的話，就不會造成重傳的現象。

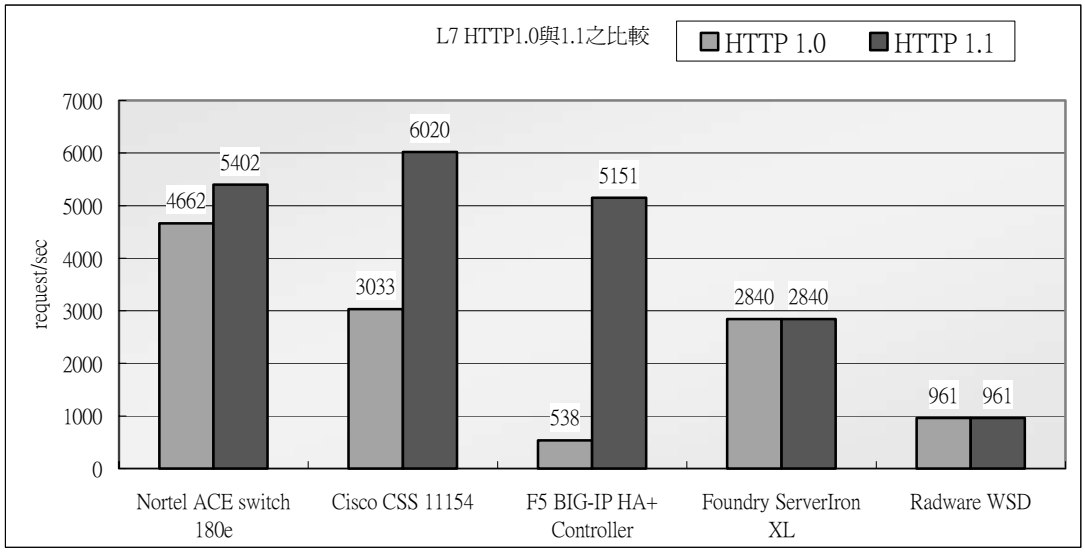


圖 9. HTTP1.0 與 1.1 效能比較

Foundry 和 Radware 兩家產品降級成 HTTP 1.0 的方式均導致我們的測試環境無法測量(Foundry 的做法會讓伺服器回傳 HTTP 1.0 的封包，而使得 WebBench 誤以爲伺服器不支援 HTTP 1.1 而無法量測，而 Radware 因 WebBench 送出的 HTTP 要求中不帶有 Keep-alive 而無法進行降級)，而不降級的話就無法正確的做 URL switching 而使得數據變得沒有意義。根據這二家使用的演算我們可以推測它們的效能大概和 HTTP1.0 差不多，甚至可能會更差！因爲他們得要去改寫 HTTP header 的內容。在圖九中我們以在 HTTP 1.0 的情況下量得的數據做爲這兩家產品的數據。在真正的瀏覽器中並不會有上述的問題。

5.5 Rule 個數對效能的影響

由於 content rules 都是字串比對的演算法，這是很花 CPU time 的演算法，所以我們使用大量的 content rules 來測試各家的產品，看看那些產品會因爲 rules 的增加而造成效能上的低落。圖十爲其結果，其中數字爲 0 的產品是測出來的數據無法參考，原因如上所述。

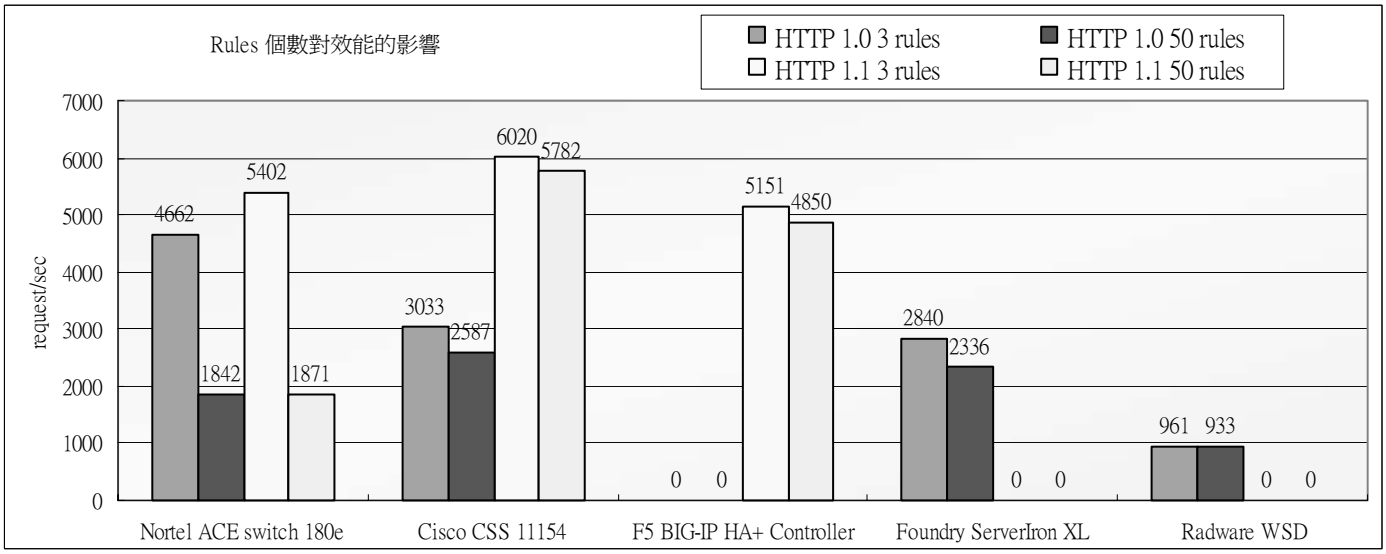


圖 10. Rules 個數對效能的影響

爲了比較上的單純，我們比較在 HTTP1.0 時的效能，因爲各家在 HTTP1.0 的作法比較相近，其中因爲 F5 產品在 HTTP1.0 的效能有問題，所以取其 HTTP1.1 和 Cisco 在 HTTP1.1 時的效能做比較。我們計算下降的百分比，結果於圖十一。

不難看出，其中 Nortel 下降的幅度是最大的，推測應當跟它規則設定的方式有很大的關係；

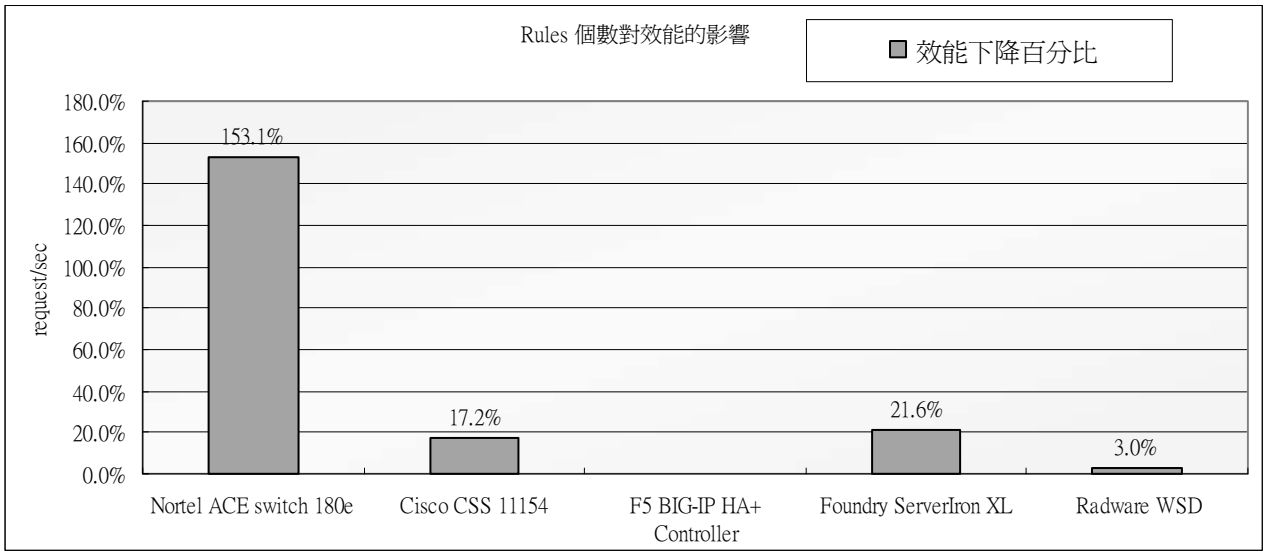


圖 11. Rules 個數增加時各家效能下降的百分比

當收到 HTTP 要求時，要去檢查每個伺服器設定的規則，如果有五十條規則，則對每個伺服器都要去檢查這五十條規則，如果有 14 台伺服器的話就必需要檢查 700 條 rules，必須要把這些規則都檢查完之後，才知道有哪些伺服器可以分派，所以才會造成效能的低落，而且這樣的作法非常

的不 scalable。比較 Cisco 和 Big-IP 在 HTTP1.1 時候的下降，Cisco 下降了 4.1%而 Big-IP 下降了 6.2%，所以 Cisco 在這方面做的比 Big-IP 來得好，大約和 Foundry 差不多。

5.6 啟動 cookie persistence 對效能的影響

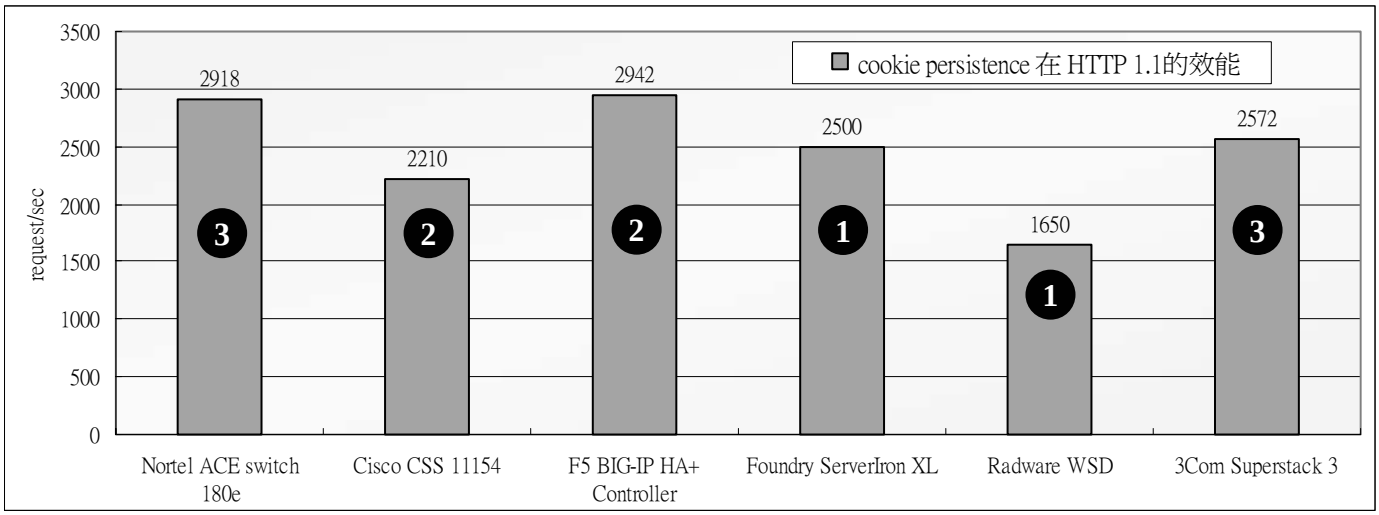
爲了支援動態網頁程式(ASP、JSP、PHP 等)的 session 功能，各家對 cookie persistence 各有不同的作法，列述三種比較常見的方法：

事先溝通：這個方法就是網頁設計者必需要自行在網頁中加入一些 set-cookie 的指令，比如 ServerA 裡面的網頁會自已發一個 cookie 叫 Server=A，ServerB 會發 Server=B 的 cookie…以此類推。然後在網頁交換器上設定 rule，只要找到 Server=A 就送到 ServerA，找到 Server=B 就送到 ServerB。這是很不好的設計，因爲設定時要自已改寫所有的網頁。Foundry、Radware 等提供這個方法。

Insert cookie：這個做法就是當 server 送出回應(response)時，網頁交換器會把封包給攔下來，然後在 HTTP header 裡加入一個 set-cookie 的指令，通常是把 server 的 ip 給打上去，這樣下次用戶端的要求過來時只要 parse 這個 cookie 就知道要送到那個 ip 去了。這個作法會比第一種作法來得花時間，不過卻不用自行修改網頁，非常的方便。Nortel、Big-ip、Cisco 等提供這個方法。

Learning cookie：這個方法是在網頁交換器設定要根據那個 cookie name 來 learning，然後當 server 送出回應時，網頁交換器會去尋找那個 cookie name 並把後面的 cookie value 給學下來，並建立一個 cookie value 與 server ip 的對照表。這個作法擁有不用修改網頁的好處，也不像第二個作法必需要改 HTTP header，應該是個很不錯的作法。Nortel、3COM 提供這個方法。

由於每一家並非擁有相同的方法，所以我們測試時選用的準則以不用自己修改網頁的方法爲主，所以大部分選用 Insert、Learning 的方法，只有 Foundry、Radware 使用事先溝通的方法。結



(長條中的數字代表使用第幾個方法。)
圖 12. Cookie persistence 對效能的影響

果如圖十二，❶代表使用第一種方法，以此類推。如果用相同的方法的產品互相比較的話，不難看出效能比較好的產品使用的 CPU 等級比較快。在這裡 Nortel、F5 表現的很不錯，不過在 L7 HTTP1.1 表現的最好的 Cisco 在這裡就不是頂好的了。而且在測試過程中，當 Traffic 量一大的時候 Cisco 就會死當。另外 Cisco 還有另一個問題，它實作的 Insert cookie(Cisco 稱作 Arrowpoint cookie) 有個 bug，當 cookie insert 完之後，下一個 request 要求的是另一台 server 的內容時，它會被重導到另一台去，並且以後所有的 request 就會到那一台去，即使用戶端要求的內容又在另外一台 server 上，不過 cookie persistence 測試並沒有做 content partition 的動作，所以沒有這樣的 error。

六、結論

6.1 功能面的評分

功能面我們分安裝簡易度、功能完整性、管理簡易度、協定支援度四大項來評分，評分標準說明如下，評分結果放在表十。

1. 安裝簡易度：安裝上均不算困難。其中 Nortel 的產品因功能較多，需要花較多的時間去理解其內容。Cisco 的產品手冊以功能爲主，範例多以功能爲主，缺少完整的設定範列，不過也會內建的 script 可以做快速的設定，但學會設定得花一段時間。F5 的產品在初始化設定後，它的 web 介面就非常的友善，也有精靈能夠教人做一步步的設定。Foundry 的產品除了一開始舊版的 firmware 並沒有提供 IP routing 的功能，使得我們跟代理商索取新版的 firmware 自己更新外，其餘的在安裝設定上並不難。Radware 的 ConfigWare 設定環境相當的直覺好用，但由於一開始拿到的是舊版的 firmware，缺乏如支援 cookie 等功能，也一樣經由自行更新 firmware 才獲得解決。3Com 的產品因一開始未注意其 console cable 需用跳過線的，花了不少的時間；又因預設安裝的 Java Runtime Environment (JRE) 1.3.0_02 版在 IE 5.0 或 NetScape 4.72 上都無法使用(瀏覽器會不正常結束)，最後自行安裝新版的 JDK 1.3.1 才獲得解決。但 web 管理介面安裝完全後，整個介面十分直覺，在設定上很好上手。

2. 功能完整性：這方面我們主要是以 L4、L7、容錯、其他特殊功能，如頻寬管理，VPN 及 firewall 的負載平衡等功能做評估。Nortel、Cisco、F5、Foundry 的產品在 L7 功能均相當完整，儘管 Cisco 及 3Com 並不支援 direct routing，但 direct routing 並非 L7 產品的特點，所以我們仍然給予很高的評價。此外，各家亦有其獨特的功能，如 Nortel 的產品可以 parse 跨過多個 frame 的 URL，Cisco 有 content replication 等特有功能。有的如 3Com 雖缺乏 L7 中的 URL switching 這一大項，但其訴求爲低價位的市場，對於不需要 URL switching 的單位來說，因爲價格最爲低廉，亦相當具有競爭力。

3. 管理簡易度：這六家產品當中， Cisco、F5、ServerIron 三家的產品不管是在命令列的操作方式或是 Web 的管理介面，都做的不錯。雖然 Nortel 的 180e 的命令列方式設計的不錯，但因爲缺乏 Web 的管理介面，所以這地方我們給分略爲低些。而 Radware 及 3Com

兩家產品在命令列管理的部分，指令的使用感覺不是很直覺，亦較難了解。但是其 web 管理介面做的很好，我們仍然給予不錯的評價。

4. 協定支援度：其中 F5 對 HTTP 1.1 的支援表現的最好、cookie 的應用也考慮的很周詳，所以我們在這方面給予最高的評價。而 ServerIron 和 Radware 在 HTTP 1.1 直接降級的做法並非很好，所以我們的給予的評價較低。而 3Com 並未處理這方面的協定，所以給予的分數也是較低。

功能面的評分

產品名稱	安裝簡易度	功能完整性	管理簡易度	協定支援度
Nortel ACE switch 180e	★★★★	★★★★★	★★★★	★★★★
Cisco CSS 11154	★★★★	★★★★★	★★★★★	★★★
F5 BIG-IP	★★★★★	★★★★★	★★★★★	★★★★★
Foundry ServerIron XL	★★★★	★★★★★	★★★★★	★★★
Radware WSD	★★★★★	★★★★	★★★★	★★★
3Com Super stack 3	★★★★	★★★	★★★★	★★★

表 8. 功能面的評分

效能面的評分

產品名稱	NAT 負載平衡	URL 負載平衡 (HTTP 1.0)	URL 負載平衡 (HTTP 1.1)	Cookie persistence
Nortel ACE switch 180e	★★★★★	★★★★	★★★★	★★★★★
Cisco CSS 11154	★★★★	★★★★	★★★★★	★★★
F5 BIG-IP	★★★	★★	★★★★	★★★★★
Foundry ServerIron XL	★★★★★	★★★★	★★★	★★★★
Radware WSD	★★	★★★	★★	★★★
3Com Super stack 3	★★★★★	不支援	不支援	★★★★

表 9. 效能面的評分

整體評價

我們分別以功能面及效能面兩方面的總分給予功能面及效能面的評價，另外，價格也是服務廠商在選購產品時的一個重要參考，我們一併附上價格方面的評分。

產品名稱	功能面總評	效能面總評	價格	總分
Nortel ACE switch 180e	★★★★	★★★★★	NT$ 1012K	12
Cisco CSS 11154	★★★★★	★★★★	NT$ 1250K	12
F5 BIG-IP	★★★★★	★★★	NT$ 837K	12
Foundry ServerIron XL	★★★★	★★★★	NT$ 374K	13
Radware WSD	★★★	★★	NT$ 1095K	8
3Com Super stack 3	★★	★★★	NT$ 276K	10

表 10. 產品總評分表

6.2 測試感想

在這次測試中，我們發現 L4 部分由於各家的產品效能幾乎都很好，儘管我們不斷的增加測試平台的 PC 數量試圖達到產品的能力，但發現實在無法辦到。在分析 L4 測試的 throughput 時，我們發現 HTTP 回應的流量高達 700 多 Mb/s，幾乎達到八台伺服器所有 link 速度的總和，即使增加 PC 也沒有在測得的數據上並沒有減緩的趨勢。因此我們認爲往後要測試相關產品，測試平台至少要能處理 1Gb/s 以上的流量，否則無法測出待測物真正的能力。

儘管 L7 的封包處理使得在管理上較具彈性，而且可以解決一些需要固定伺服器的應用，如 cookie 及 SSL 等，但由於會使效能降低，所以網管人員在決定是否使用 L7 的功能時應注意到這個問題，同時做效能及功能的權衡。對設備及晶片製作廠商而言，輔助 L7 功能的 ASIC 或網路處理器是個可以研究發展的目標，而且事實上目前已經有廠商，如 PMC Sierra 已開發可處理 L7 相關功能的網路處理器[15]，這方面的產品同時也可以做爲學術界研究的目標。

這次借測的產品全部都是由國外廠商開發，國內的交換器產品目前有做到 L4 以上的似乎還沒有，更不要說 L7 的部份。不過值得慶幸的是在這些產品中並沒有很突出的一二家“大廠”，每家的產品都在各方面有很好的表現，不過在別的方面就相形見絀了，而且有幾家的產品還不是很成熟，都帶有“小小的”bug，所以有興趣朝這方面產品開發的設備廠商還是很有機會變成“大廠”的，臺灣加油！

七、參考文件

[1] 張智晴、林盈達, “伺服器叢集：技術與系統解構(上)”，*網路通訊*，117 期，62-67 頁，2001 年 4 月。

[2] V. Pai, M. Aron, M. Svendsen, G. Banga, P. Druschel, W. Zwaenepoel, and E. Nahum, “Locality-aware request distribution in cluster-based network servers,” *Proceedings of the 8th International Conference on Architectural Support for Programming Languages and Operating Systems*, October 1998.

[3] T. Dierks, “The TLS Protocol Version 1.0” *RFC 2246*, January 1999.

[4] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, Hypertext Transfer Protocol -- HTTP/1.1,” *RFC 2616*, June 1999.

[5] Mohit Aron, Peter Druschel, Willy Zwaenepoel, “Efficient Support for P-HTTP in Cluster-Based Web Servers”, *USENIX 1999 Annual Technical Conference*, Monterey, CA, June 1999.

[6] Brendel et al., “World-wide-web server with delayed resource-binding for resource-based load balancing on a distributed multi-node network” *United States Patent* 5,774,660.

[7] http://brahms.cis.nctu.edu.tw/web_switch/product/.
[8] http://www.nortelnetworks.com/products/01/alt180/index.html.
[9] http://www.cisco.com/univercd/cc/td/doc/pcat/11000.htm.
[10] http://www.f5.com/f5products/index.html.
[11] http://www.foundrynet.com/products/webswitches/serveriron/index.html.
[12] http://www.radware.com/content/products/wsd.htm.
[13] http://www.3com.com/products/en_US/prodlist.jsp?tab=cat&pathtype=purchase.
[14] C. Edward Chow and Weihong Wang, "Design and Implementation of a Linux-based Content switch," *CCL document*, page 11.
[15] http://www.pmc-sierra.com/products/details/pm2329/.

附錄一、測試用 PC 之硬體規格

項目	規格
中央處理器	Intel Pentium !!! 1 GHz
主機板	MSI MS-6215 BOOKSIZE 815e
記憶體	Kingmax RAM 256 MB
硬碟	IBM IC35L020 20 GB HD
網路卡	Intel PILA8460C3, 10/100M

附錄二、各代理商連結網頁

代理商	原廠	連結網頁
豪勉科技	Nortel	http://www.hauman.com.tw
聚碩科技	Cisco	http://www.sysage.com.tw
F5 台灣分公司	F5	http://www.f5.com
懇懋科技	Foundry	http://www.phitech.com.tw
寅騰科技	Radware	http://www.welton.com.tw
3Com 台灣分公司	3Com	http://www.tw.3com.com

附錄三、相關測試工具連結網頁

測試工具	連結網頁
WebBench	http://etestinglabs.com/benchmarks/webbench/webbench.asp
Sniffer	http://www.sniffer.com.

附錄四、**WebBench workload file set**

Class	百分比	**Class**	百分比
CLASS_223.gif	20	CLASS_6040.htm	14
CLASS_735.gif	8	CLASS_11426.htm	16
CLASS_1522.gif	12	CLASS_22132.htm	7
CLASS_2895.gif	20	CLASS_404	2

(class_223.gif 代表的是每個檔案大小爲 233byte 的 GIF 圖檔，佔所有要求的 20%。Class_404 裡面的要求是不存在的。)