快取伺服器之比較、追蹤與評估

投稿領域：網路伺服器規劃

林逸祥 林盈達

國立交通大學資訊科學研究所

300 新竹市大學路 1001 號

TEL:(03)5712121 EXT.56667

FAX:(03)5712121 EXT.59263

E-mail: einstein@chu.edu.tw , ydlin@cis.nctu.edu.tw

主要聯絡人: 林逸祥

摘要

隨著全球資訊網(World Wild Web)的持續成長，為了減少網路頻寬的浪費、降低伺服器負荷、加快瀏覽網頁時的反應速度，使用代理(proxy)快取(caching)伺服器是近年來的趨勢。在本篇文章裡，將比較目前主要的快取伺服器產品，分析評比報告；並且剖析目前普遍使用開放原始碼(Open Source)的 squid[1]快取程式的運作以及程式架構，最後測試 squid 在 FreeBSD [2]、Linux[3]、 Solaris [4]使用三種不同磁碟快取目錄的存取下之效能及穩定度。經過測試發現：在高網頁需求存取量時，因磁碟存取較為頻繁，若沒有選擇對該系統較佳的存取方法，將造成極大的差別。測試結果證明，在我們的測試環境裡，使用 FreeBSD 配合 diskd 的磁碟存取方式，最高可處理每秒 570 個網頁需求，且平均反應時間只有 6.4 秒，為最佳的組合方式。

關鍵字: 快取, 代理, 伺服器, 效能, squid, proxy, caching, Solaris, Linux, FreeBSD

1. 簡介

所謂「快取(caching)」，就字面來看即「快速取得」的意思，常常有人將其與「代理 proxy」的意義混淆，其實這是不同的兩件事。「快取」主要意思就是將經常會使用到的物品，放置在很方便又快速取得的地方，用以節省時間增加效率。而「代理」是指代替某人做某事。

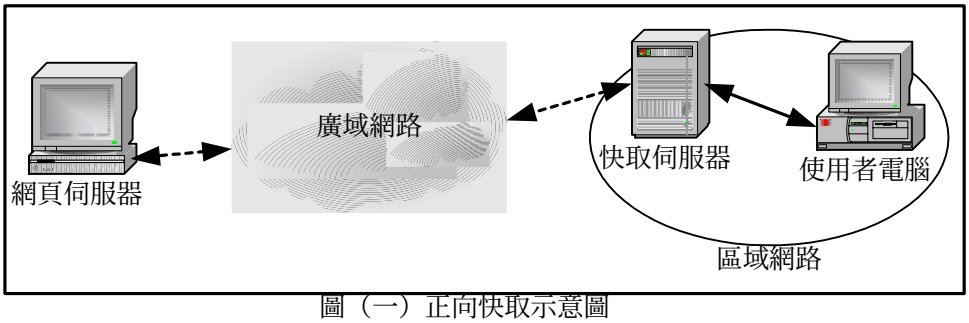
在平常生活中，我們會把最近常會使用到的書、物品等，從書架上或物品原本擺放的地方，搬移放置在書桌上方便取用，此書桌即可視為一個快取系統。而在電腦的世界裡，也大量利用了快取技術來增強電腦的效能，目前幾乎所有的電腦設備都有快取，從中央處理器(CPU)的內外部快取、介於中央處理器與主記憶體中間的快取記憶體，顯示卡、硬碟、光碟燒錄機、網路卡也都有內建快取，這些皆是屬於硬體的快取；而在全球資訊網的網頁存取上也應用了快取技術，從網頁瀏覽器內建的個人使用快取，又因為在同個區域網路裡瀏覽相同網頁的機率較高的特性，且專線費用昂貴，因而衍生出在網路上建置多人共享的快取伺服器。

一般而言，使用快取伺服器有三大優點：一、減少頻寬浪費；二、降低網頁伺服器的負擔；三、使用者可較順暢快速瀏覽網頁。 但是快取伺服器也會發生幾項問題，如不能保證所看到的網頁是最新的，設定不當可能耗費更長等待時間。

依快取伺服器擺設的相對位置、扮演的角色以及設定上的差異，通常將其分為下列三類：

一、使用者端的正向快取(forward caching)，如圖（一）。

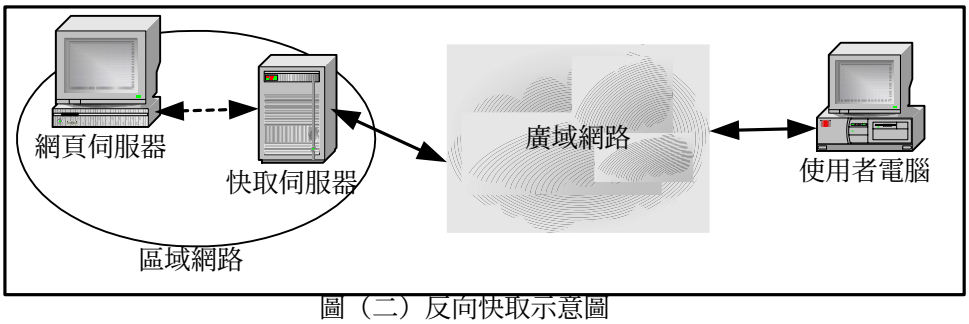
現今的網頁瀏覽器皆普遍支援設定代理快取伺服器，即是使用這樣的架構。



圖（一）正向快取示意圖

二、網頁伺服器端的反向快取(reverse caching)，如圖（二）。

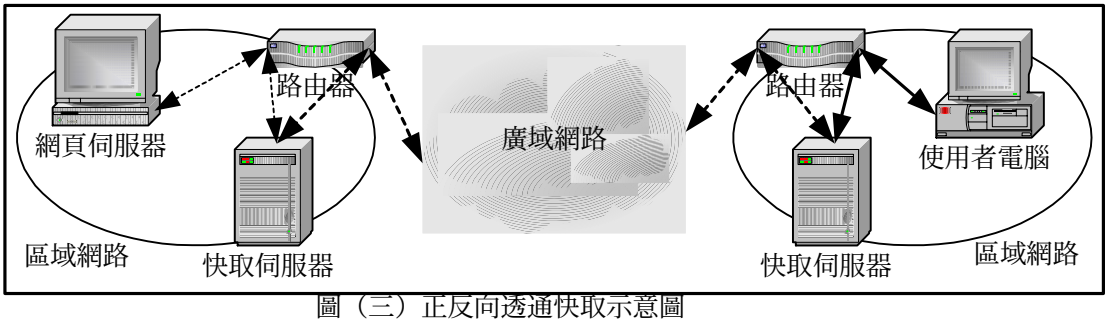
在這種架構下，內部網站網址查詢的 IP 地址皆指向快取伺服器，所以使用者將直接與快取伺服器連線，若快取伺服器無所要網頁，再依照網站名稱查詢內部對應 IP 地址，才到真正的網頁伺服器抓取資料。



圖（二）反向快取示意圖

三、在網路節點上的透通快取(transparent caching)，如圖（三）。

所謂透通快取設定就是說：使用者端的瀏覽器或網頁伺服器根本不用做額外的設定，即可使用快取伺服器，主要原理是由支援網路傳輸層的路由器或網頁交換器等設備，將使用者的網頁需求導向到快取伺服器，在使用上，使用者將以為直接跟網頁伺服器連結。圖（三）做了兩階段的透通快取：在使用者端稱為正向透通快取，在網頁伺服器端稱為反向透通快取。



圖（三）正反向透通快取示意圖

在探討快取系統的議題方面，主要分為下列幾類[5]：

一、網頁傳輸方面：包括網頁伺服器的效能與特性、代理快取伺服器的流量特性、使用者對網頁伺服器的感覺、網頁瀏覽器的流量特性、代理伺服器之間的通訊協定。

二、快取內容方面：網頁快取內容的一致性、網頁快取內容的替換方法、預先取得快取的方法、與 HTTP 機制相關的研究。

三、其他議題及法律上的問題。

雖然快取產品種類繁多，不過如果依照其快取引擎，可加以分類整理如表(一)[6]。

快取引擎	快取產品		快取引擎	快取產品
Apache	Apache Proxy Module		Kotetu	NAIST-1/2
Aratech	Aratech(JAGUAR)-2000		Lucent	Lucent-50/100/100z WebCache
CacheEngine	Cisco Cache Engine 500/7000 series		Microsoft	Microsoft ISA
CacheOS	CacheFlow Server/Client Accelerator		MOWS(java)	MOWS(open source)
Cachier	Cachier(dynamic content delivery)		Netscape	Netscape proxy server
Inktomi	Inktomi, Compaq, Dell, F5-EDGE-FX		Novell/Volera	IBM, Microbits, Stratachche, Excelerator
Jigsaw	W3C’s JAVAServer		NetApp	NetApp-C1105/C6100
Imimic	iMimic, Cintel		Squid	Squid-2.4.STABLE1, Swell-1450, Cobalt,WebSpeed

表（一）快取產品的分類（依照快取引擎）

而在上表眾多的快取產品中，開放原始碼且免費的 squid 代理快取程式，是功能強大且最多人使用的快取軟體，其支援了全功能的網頁代理快取（包括 HTTP、FTP 及其他 URL），且可以代理 SSL 的網頁及本身快取 DNS 查詢，聯合數台 squid 快取伺服器可建構出快取架構，支援了 ICP[7]、HTCP[8]、CARP[9]、WCCP[10]數種快取通訊協定， 也可用 cachemgr.cgi 透過網頁或利用 SNMP 做即時快取狀況查詢。

2. 市面上的快取產品

2.1 產品比較

茲依照快取產品的支援功能，將目前市面上幾種比較常見的產品做簡單比較，如表(二)。

	CacheFlow 6000[12]	Cisco Cache Engine 500[13]	Compaq[14] TaskSmart-C4000	Squid- 2.4-STABLE1
一般通訊協定支援	HTTP 1.0/1.1 SSL/DNS	僅 HTTP 1.0/1.1	HTTP/FTP/SSL DNS	HTTP/FTP/SSL/DNS
快取通訊協定支援	WCCP v1/v2	ICP/WCCP/ MHSRP	ICP/WCCP/ WPAD	ICP/WCCP/CARP
管理支援	SNMP/Web/ssh	SNMP/Web/telnet	SNMP	SNMP/Web/telnet/ssh
認證機制	LDAP/RADIUS	RADIUS	LDAP	LDAP/MSNT/NCSA PAM/SMB/YP 等模組
影音串流 (Streaming)	MMS/RTP/RDP/ MP3/ FLASH	無	MMS/RTP/RDP	無

表（二）市面上常見幾款快取產品支援功能比較

在表（二）中，SSL 是指可以透過其進行 SSL 加密通訊傳輸；認證機制是指用哪些方法檢查確認可使用的帳號；影音串流則是指支援哪些即時的多媒體傳輸協定。

2.2 評估報告

2.2.1 評估項目

TMF(The Measurement Factory)[15]從 1999 年開始至今已舉辦過三場名為 CachOff[16]的快取產品評估測試，今年九月將會舉辦第四場評比。底下為第三場 CacheOff 針對 31 項產品的主要的評比項目及簡單說明：

1. 價格：列出各產品的價格，為一般購買伺服器的重要考量點。
2. 尖峰處理能力：測驗快取伺服器可處理每秒網頁需求數的極限值。
3. 網頁回應時間：包括擊中(所要求的網頁在快取空間內)、未擊中(所要求的網頁不在快取空間內，而要透過網路連結至網頁伺服器讀取)以及平均回應（指所有網頁需求）的時間 。

4. 節省比率(擊中率及時間)：擊中率是指所有網頁需求可直接在快取空間取得的比率；時間節省比率是指使用快取比不使用快取的回應時間的比率。
5. 一千塊美金的價值(每秒擊中次數/每秒回應次數) ：將擊中次數或回應次數除以該產品的價格，可以公平比較何者價值最高。
6. 跳電後開機第一次擊中/未擊中需要幾分鐘 ：比較各伺服器的穩定性及容錯恢復能力。
7. 快取內容的保存時間：測試使用尖峰流量填滿快取空間的時間。

2.2.2 評估結論

分析第三場 Cache-Off 的評比報告的結果，整理各項目最優劣的產品比較如表（三）：

評估項目			最優(上)及最劣(下)的產品	數值
1	價格(美金)		Stratacache-D	$784
			NetApp-C6100	$101,700
2	尖峰處理能力(每秒最多可處理網頁需求數)		Dell-200x4	3310
			Microbits-P	115
3	網頁回應時間(秒)	擊中	IBM-230, iMimic-Alpha, Lucent-50	0.03
			Stratacache-D	0.60
		平均	Lucent-50	1.36
			IBM-230	1.99
		未擊中	Compaq,F5-EDGE-FX,IBM-220	2.77
			Squid-2.4.D4	3.51
4	節省比率(%)	命中率	Aratech-2000	56.1
			IBM-230	30.8
		時間	Lucent-50	51.4
			IBM-230	29.0
5	一千塊美金的價值	每秒擊中次數	Microsoft-2	78
			Lucent-50, NetApp-C6100	11
		每秒回應次數	Microsoft-2	145
			Lucent-50	20
6	第一次需要幾分鐘	未擊中	Lucent-50	1.3
			Compaq-C2500	6.6
		擊中	Lucent-50	1.3
			Compaq-C2500	6.6
7	快取內容的保存時間(小時)		Squid-2.4.D4	24.4
			IBM-230	2.1

表（三）第三場 Cache-Off 的評比結果優劣產品比較

觀察表（三）的評比結果之後，我們可以發現有兩款產品的上榜率最高且成兩極化反應：一為 IBM-230、另一為 Lucent-50。IBM-230 的擊中回應時間是最快的，但是平均回應時間卻最慢，另外在節省比率與快取保存時間都是最差的，因為是使用 gigabit 的網路卡，所以命中的回應時間較快，其他則是因為內部使用的快取程式在處理尖峰量傳輸下(每秒 1400 個網頁需求)，造成其可配置硬碟不足的情況，因而無法計算保留太多高使用的快取。反觀 Lucent-50 因其尖峰傳輸量只有每秒 200 個網頁需求，且價格並不便宜，所以一千塊美金的價值就較低，，且架構較為簡單，硬碟空間較大，因此平均回應時間較快，較節省時間，開機回應也比較短。

此外，觀察評比結果在尖峰傳輸量項目超過每秒 2000 個網頁需求的各項產品，共通點是其記憶體皆大於 2GB 且網路卡頻寬是 gigabit 以上，當然價格也很高。由此可知，記憶體及網路卡是提高快取伺服器處理效能的兩項重要配備。

3. 透析開放原始碼的 squid

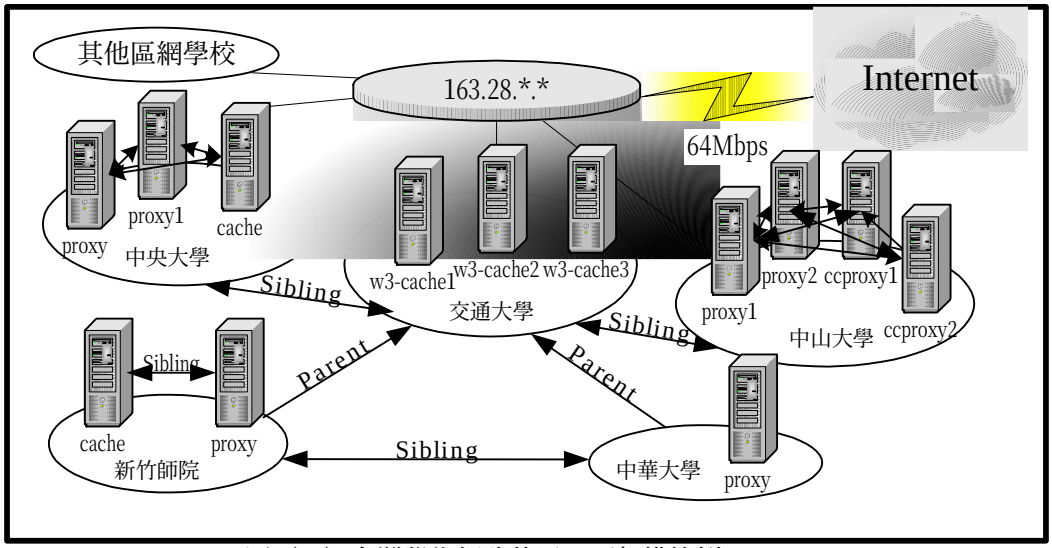
3.1 squid 的簡介

Squid 程式源自於 ARPA[17]贊助的 Harvest[18]計畫，是由一群沒有支薪或有支薪的志願者貢獻設計，使用 GPL 的版權宣告，是免費且開放原始碼的程式，主要在 UNIX 平台上發展，目標是要成為支援完整、高效能且穩定的快取程式，所有的網頁需求皆只由一支 non-blocking、I/O driven 的程式（名稱即為 squid）控制，而且程式版本的更新頻率很高，可以針對目前的環境需求加以改良，具有很大的擴充性，因此為最人多使用的快取伺服器軟體。Squid 的網址為 http://www.squid-cache.org。

3.2 使用 squid 的設定範例

3.2.1 台灣學術網路使用 squid 架構的範例

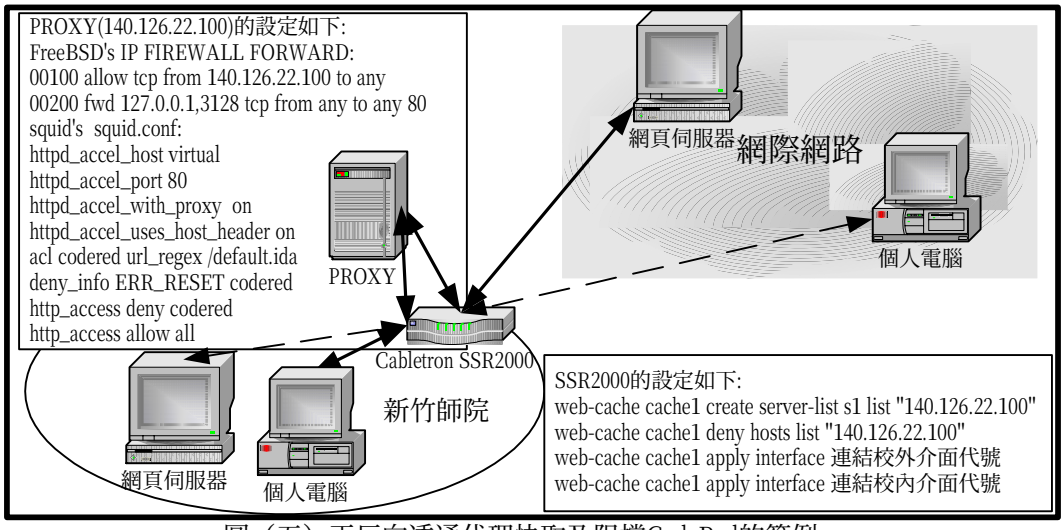
台灣學術網路目前大部分學校皆是採用 squid 架構快取伺服器[19]，由於區網中心可直接使用到出國專用 64Mbps 的 proxy 保留頻寬，相互設為 sibling 可利用彼此已有的快取內容，而連接區網中心的學校，則需設定區網中心提供的快取伺服器為 parent，可間接使用到專用頻寬，達到管制及節省頻寬流量的目的。主要架構如圖（四）。



圖（四）台灣學術網路使用squid架構範例

3.2.2 使用 squid 配合路由器(router)做正反向透通性代理快取及過濾網頁的範例

在撰寫此篇文章其間，全球網路正被攻擊 IIS 網頁伺服器漏洞的 CodeRed[20]病蟲所侵襲，為了有效阻擋其攻擊，此範例為實際在新竹師院的路由器設定正反向透通快取，將所有通過的網頁流量，導向 squid 的代理快取伺服器過濾，再加上存取控制(ACL)，可以有效地阻擋 CodeRed 的攻擊。簡單的設定範例如圖（五）。



圖（五）正反向透通代理快取及阻擋CodeRed的範例

3.3 squid 支援的通訊協定

Squid 除了支援 HTTP、FTP、gopher、WAIS 通訊協定的代理快取之外，本身還做網域名稱的快取用以加快網址及其 IP 位址的轉換，此外，也支援 SSL 網頁 的代理功能。

在代理伺服器相互之間溝通的協定中，支援 ICP、HTCP、CARP、WCCP。

3.4 程式架構

Squid 最新的穩定版本 2.4-STABLE1 的程式碼，包含函式庫、標頭檔已經高達七萬餘行，是一個功能強大的程式，列舉其重要元件及說明如表（四）[21]。

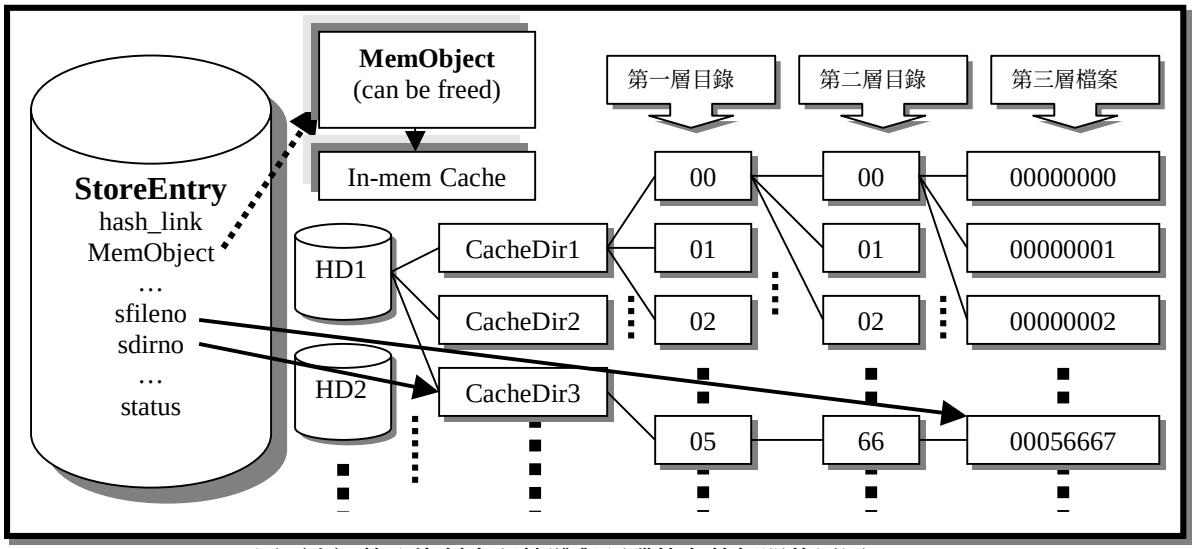
主要元件	功能說明
使用者端(Client Side)	接受、分析及執行使用者的網頁需求
伺服器端(Server Side)	轉送未擊中的網頁需求到其他合作的伺服器
儲存管理(Storage Manager)	緊密連結使用者端及伺服器端的網頁需求
轉送需求(Request Forwarding)	轉送網頁需求
節點選擇(Peer Selection)	依照設定負責選擇使用最佳相鄰的快取伺服器
存取控制(Access Control)	關於快取資料需求的來源及目標存取權限控制
認證架構(Authentication Framework)	負責處理一些需要通過認證使用的網頁需求
網路通訊(Network Communication)	負責 TCP 與 UDP 的通訊處理
硬碟存取(File/Disk I/O)	負責快取物件的存取
相鄰快取(Neighbor Caches)	負責維護相鄰的快取列表、決定要送需求給誰
網域名稱快取(IP/FQDN Cache)	網域名稱與 IP 位址的查詢快取

網路狀況測量資料庫	量測紀錄伺服器的傳輸時間用來當節點選擇的依據
重導機制(Redirectors)	重寫(rewrite)使用端的網址需求可做額外的控制
自治區號碼(ASN)支援	使用自治區號碼(Autonomous System Numbers)做存取控制
設定檔處理功能	處理分析設定檔參數值
取回資料(Callback Data)配置	配置取回資料的空間
除錯函式	可設定層級除錯追蹤程式
錯誤訊息處理	產生不同語言的錯誤訊息
事件排程(Event Queue)	快取檔案取代、清除目錄、ICP 詢問等事件的排程
檔案描述值(FD)管理	管理追蹤被讀寫的檔案
雜湊表(hash table)的支援	產生、比對雜湊表的值
網路快取協定(ICP)	處理 RFC2186/2187 定義的 ICP 通訊協定
驗證(ident)查詢	支援 RFC931 驗證使用者名稱查詢
記憶體管理	在記憶體分配一塊區域做常用的物件管理
多點傳播(Multicast)支援	目前僅支援 ICP 查詢
持續(persistent)的伺服器連結	管理與相鄰快取伺服器或網頁伺服器持續的 http 連結
更新法則	決定快取物件是否需要更新
SNMP 支援	可利用 SNMP 協定察看快取伺服器的即時狀況
URN 支援	在 squid 支援 URN 的目標是在管理鏡射(mirror)站的列表

表（四） squid 程式的主要元件說明

3.5 squid 使用的資料結構

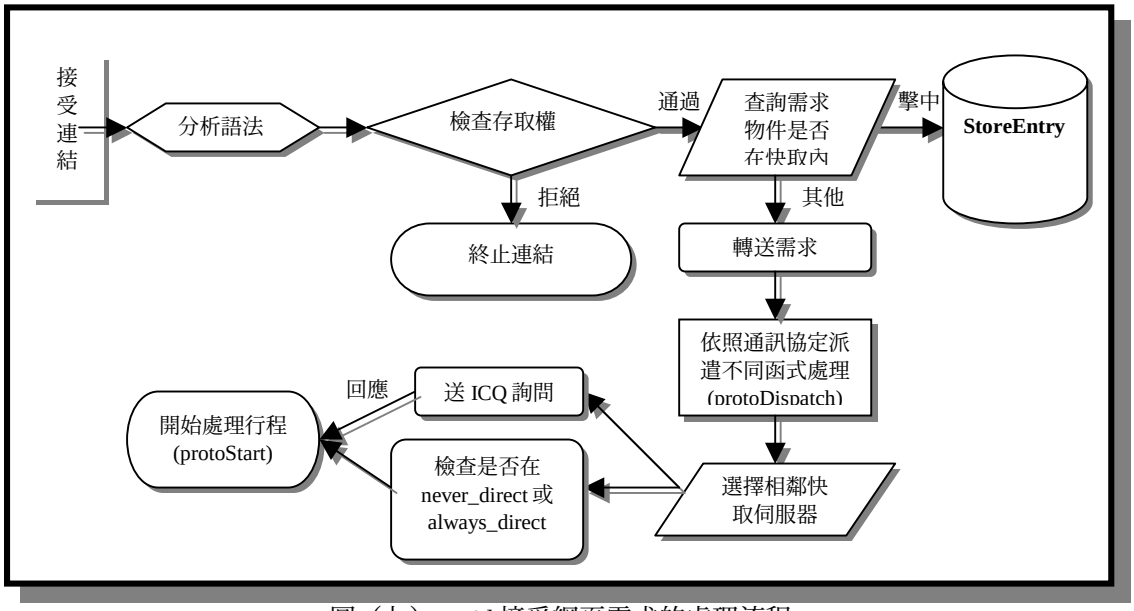
從 cachemgr.cgi 管理程式，我們可以發覺 squid 在記憶體中大約有 62 種資料結構：主要包括 2K、4K、8K、Socket 的 buffer、存取控制表(acl)、快取摘要表(CacheDigest)、用戶端網頁需求、連結狀況、網頁表頭資訊、相鄰快取資訊、網域名稱快取、、、等，以及最重要的網頁快取存放資訊與內容。圖（六）為快取資料在記憶體與硬碟中的存放相關位置，每一筆快取資料主要是由 StoreEntry 這個資料結構負責記錄，當一開始使用時，在記憶體的資料由 MemObject 負責，但如果符合被取代條件，這個資料結構則會被釋放掉，因此在圖中以虛線表示。



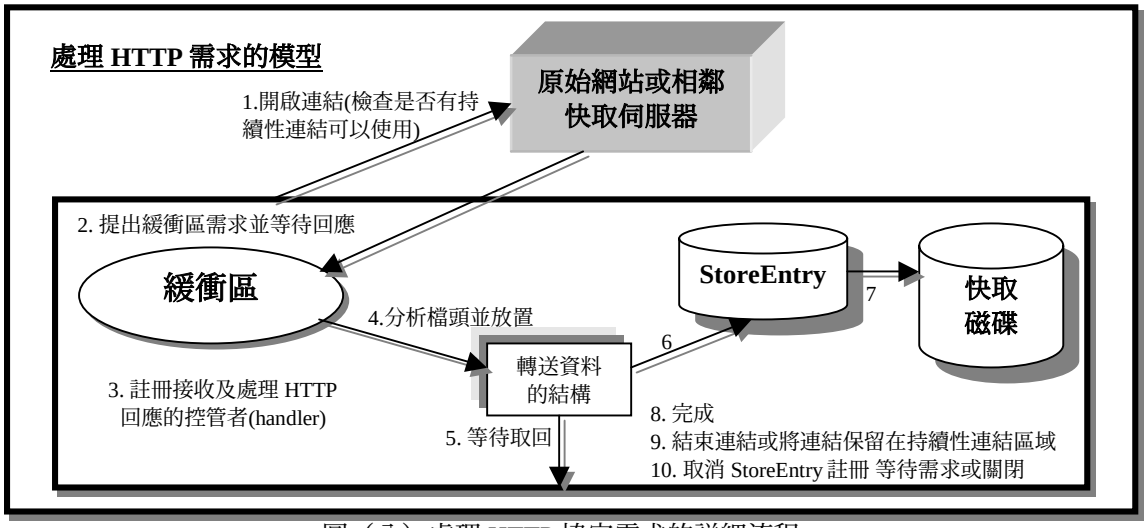
圖（六）快取資料在記憶體與硬碟的存放相關位置圖

3.6 程式流程

程式一開始執行一連串的啟動、設定檔分析、配置、測試動作後，使用 daemon 模式，當父行程產生(fork)另一子行程後，父行程進入無窮迴圈，主要在維護子行程能持續執行下去；而子行程也進入一無窮迴圈，等待處理連結需求（可使用 pool 或 select 方式）、快取空間取代、事件處理、、、等動作，在程式執行期間還可直接接受更改設定，也可接受關閉(shutdown)指令，正常終止程式。圖（七）為處理連結需求的流程，圖（八）則為接受 HTTP 協定後的詳細處理流程。[22]



圖（七）squid 接受網頁需求的處理流程



圖（八）處理 HTTP 協定需求的詳細流程

4. 測試評估 squid 的效能

4.1 先前已經對 squid 做過的效能評估

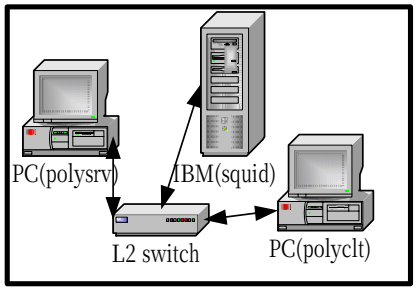
在 squid 的網站上列了一些針對 squid 程式設定的測試項目及結果，整理分析如表（五）：

種類	測試項目	測試方法	測試結果
硬碟快取空間分配相關的測試	吞吐量與快取空間大小的關係	將快取空間 1024,2048.. 每次增加 1024MB	當 cache 的空間愈大時，雖然命中率會些微升高，但是傳輸率變小，且回應的時間增長
	快取空間佔快取硬碟比率低時的效能	設定快取空間為原本的 25%	當累積的需求數多時，每秒需求數傳輸量升高，並且有爬升的趨勢
	多顆與單顆快取硬碟的差異	總快取空間分配到六顆硬碟	使用六顆硬碟時的每秒需求數傳輸量皆比使用一顆時略高些
使用函式或演算法相關的測試	使用不同記憶體配置的函式	FreeBSD 的 malloc 與 dlmalloc 的比較	使用兩種配置函式，squid 程式所佔記憶體空間差異度不大
	使用不同的快取檔案清除函式	使用 unlink 函式取代 truncate 函式	當累積的需求數多時，使用 unlink 函式的每秒需求數傳輸量略微降低
	不同的快取檔案取代演算法	使用堆積(heap)演算法的 GDSF/LFUDA 與 LRU	GDSF/LFUDA 保留較熱門的物件所以命中率較高，但 CPU 的使用率也較高

表（五）對 squid 做過的評估分析

4.2 我們的評估測試方法

前一節所提到的 squid 各項測試，皆是使用 polygraph[23]這套專門設計來測量快取伺服器效能的軟體，因此我們也沿用這套軟體來進行我們的測試，用的版本是 2.5.5。 Polygraph 主要有兩支程式：一支程式名稱為 polysrv，是用來模擬網頁伺服器；另一支程式名稱為 polyclt，是用來模擬使用者端瀏覽網頁的動作。因此至少需要三台電腦，分別模擬伺服器、使用端、架設快取伺服器，確保不會互相干擾，或造成系統資源不足，而無法正確得到我們要的結果；而且此種架構也是最接近一般實際上使用的狀況。環境設定如圖（九）、電腦及設備如表（六）。



圖（九）測試環境圖

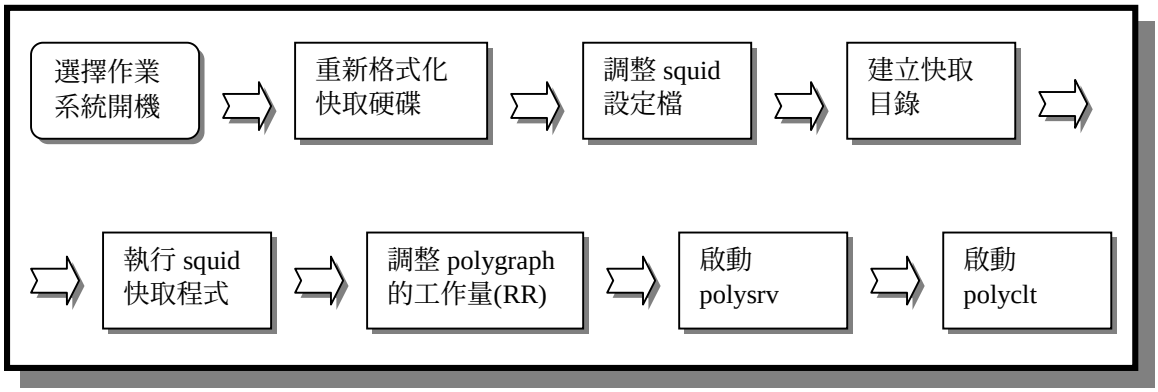
測試相關設定：

1. polygraph 使用 polymix-1 做為工作量檔，僅修改每秒需求數
2. 每顆硬碟的快取空間僅使用 500MB，兩顆共 1GB，才能在一小時內填滿快取空間，如此獲得的測試值才有意義。

名稱數量	詳細規格
PC x 2	LEO PC CPU: Pentium3-800 記憶體:256MB 硬碟容量:20GB IDE 網路卡: Intel EtherExpress pro 作業系統: FreeBSD 4.3-STABLE
PC Server x 1	IBM Netfinity 5100 CPU: Pentium733 x 2 記憶體: 1024 MB 硬碟容量: 18GB SCSI x 5 網路卡: 內建 Pcnet 作業系統: FreeBSD 4.3-STABLE RedHat Linux 7.0(kernel 2.4.8) Solaris 8 for x86
L2 Switch x 1	SMC TigerSwitch 10/100 6924M

表（六）測試設備

我們在 PC Server 的三顆硬碟分別安裝三套目前最常用作業系統的最新版本：FreeBSD 4.3-STABEL、RedHat Linux 7.0(kernel 2.4.8)、Solaris 8 for x86(2001/4/1 版本)，也調整使用系統最佳化參數、加大可用檔案描述詞(FD)以及擴大網路卡的緩衝區，並且停止了系統上一些不必要的行程（process）。剩下的兩顆硬碟則當快取硬碟。每項測試進行前皆重新格式化快取硬碟，再重新建立新的快取目錄，以確保每次測試的獨立性。測試流程如圖（十）。



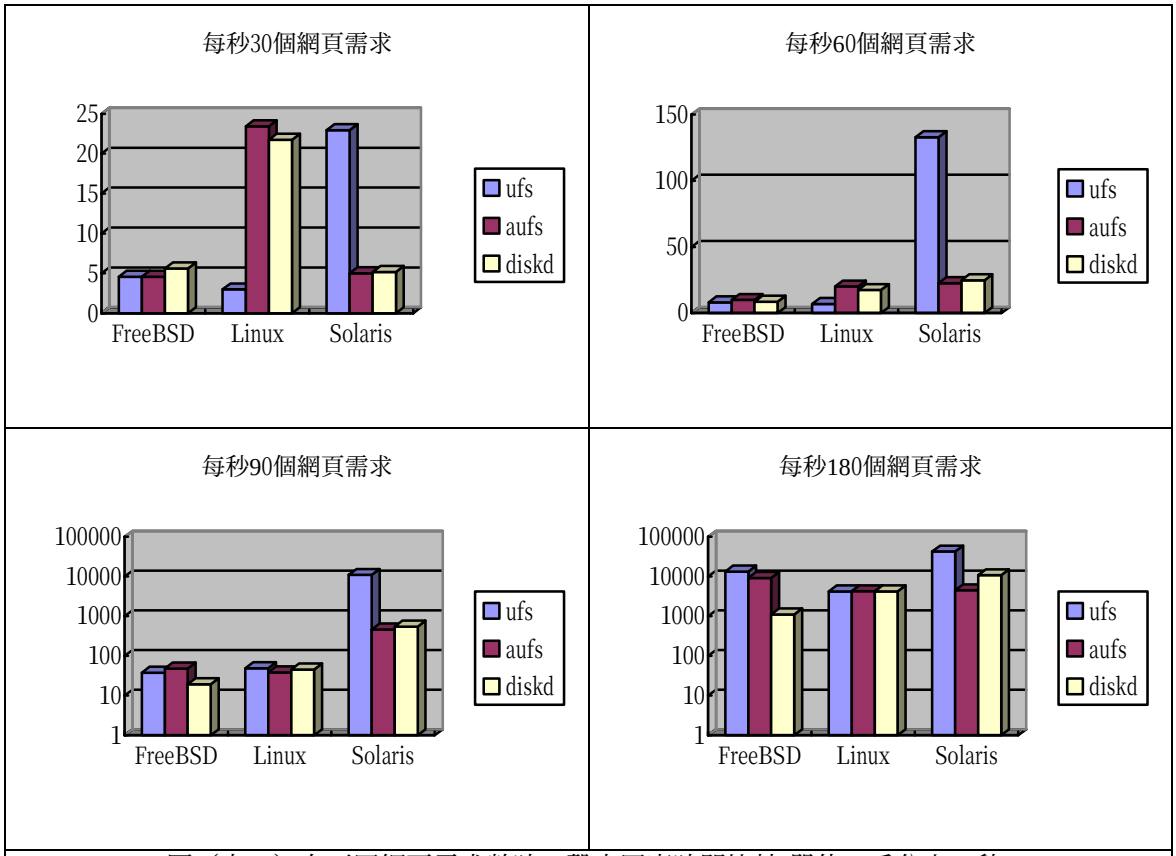
圖（十）測試流程

4.3 我們的評估測試項目及結果

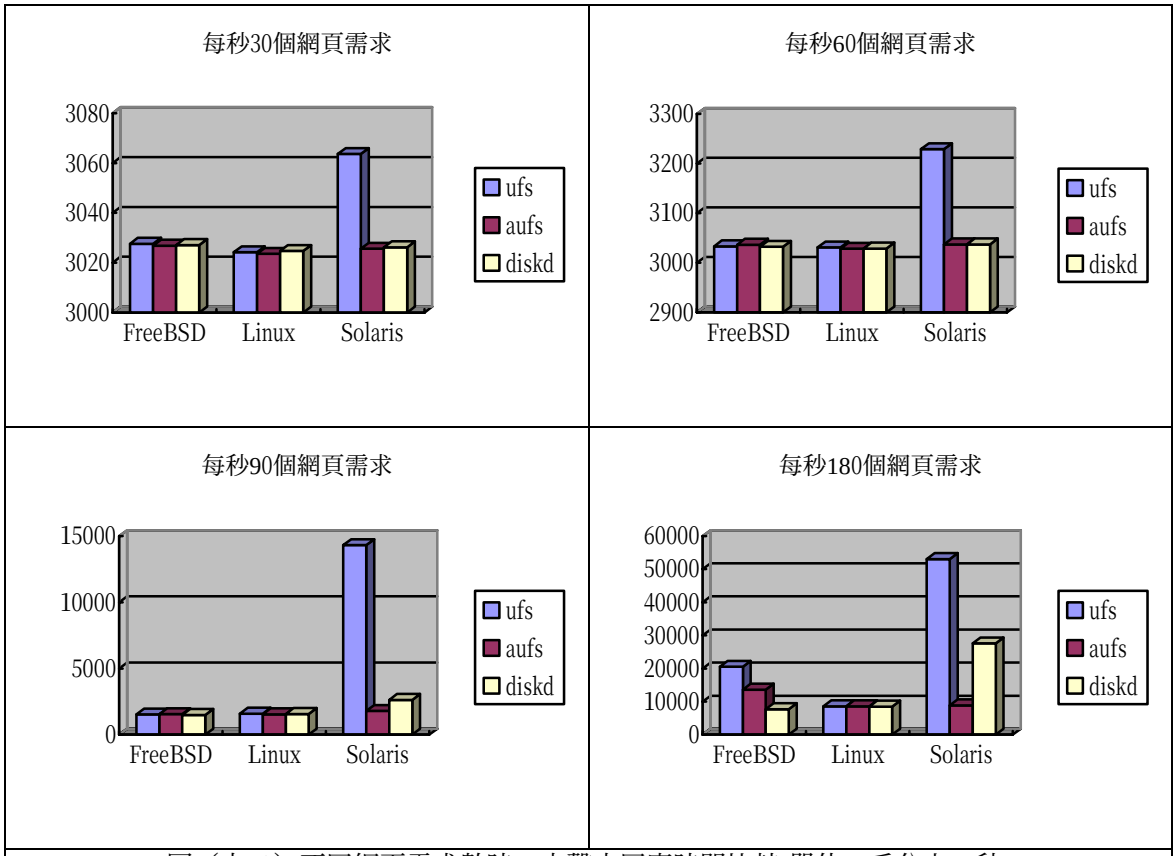
我們的測試目標是要測量出在同樣的硬體條件下，squid 快取程式在三種常用的作業系統的效能及穩定度。而影響快取系統效能主要是侷限在其硬碟存取的動作，因此我們分別使用 squid 支援的 ufs、aufs（使用多執行緒進行非同步存取，避免存取凍結）、diskd（執行額外獨立的行程進行存取，避免存取凍結）三種硬碟存取模式，調整每秒的網頁需求數，測量其擊中、未擊中及其平均回應時間、以及快取的擊中率，結果分別在圖（十一）、（十二）、（十三）、（十四）。

在圖（十一）的擊中回應時間裡，我們發現在每秒網頁需求數少於 60 時，使用 Linux+ufs 的時間是最短的，FreeBSD+ufs 則僅相差一點，但值得注意一點的是 Linux 在 aufs 及 diskd 兩種存取模式的表現卻差很多。此外，在每秒需求數大於 90 甚至到達 180 時，反應時間落差極大，因此刻度改以指數顯示，此時則以 FreeBSD+diskd 的反應時間最短。

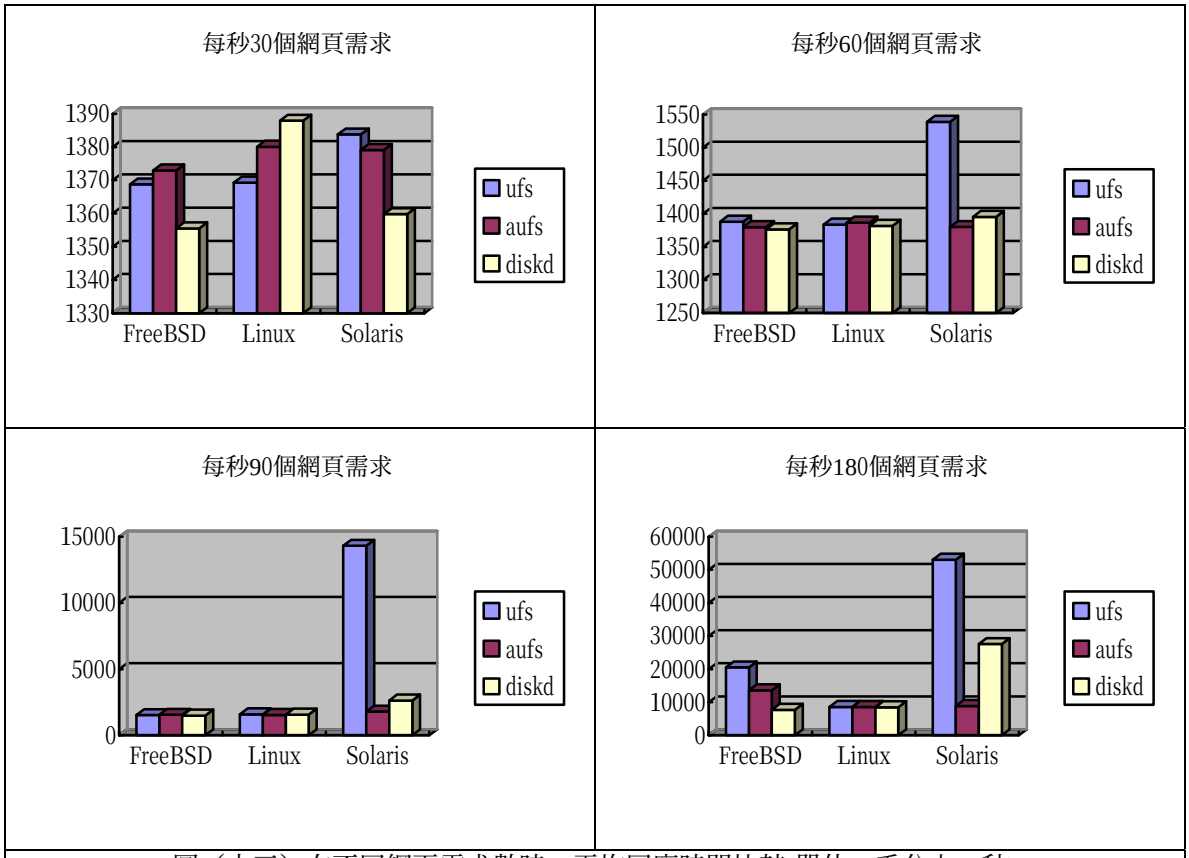
圖（十二）的未擊中反應時間與圖（十三）的平均反應時間在較低需求數時 Linux 與 FreeBSD 在各種存取模式是很接近的，而 Solaris+ufs 在到達每秒 90 個需求時，出現了極大的差距，經反覆測試仍舊如此，可以推測其 ufs 檔案系統的存取極限在 60-90 之間。此外，也可以發現 Linux 的表現是較平穩的。



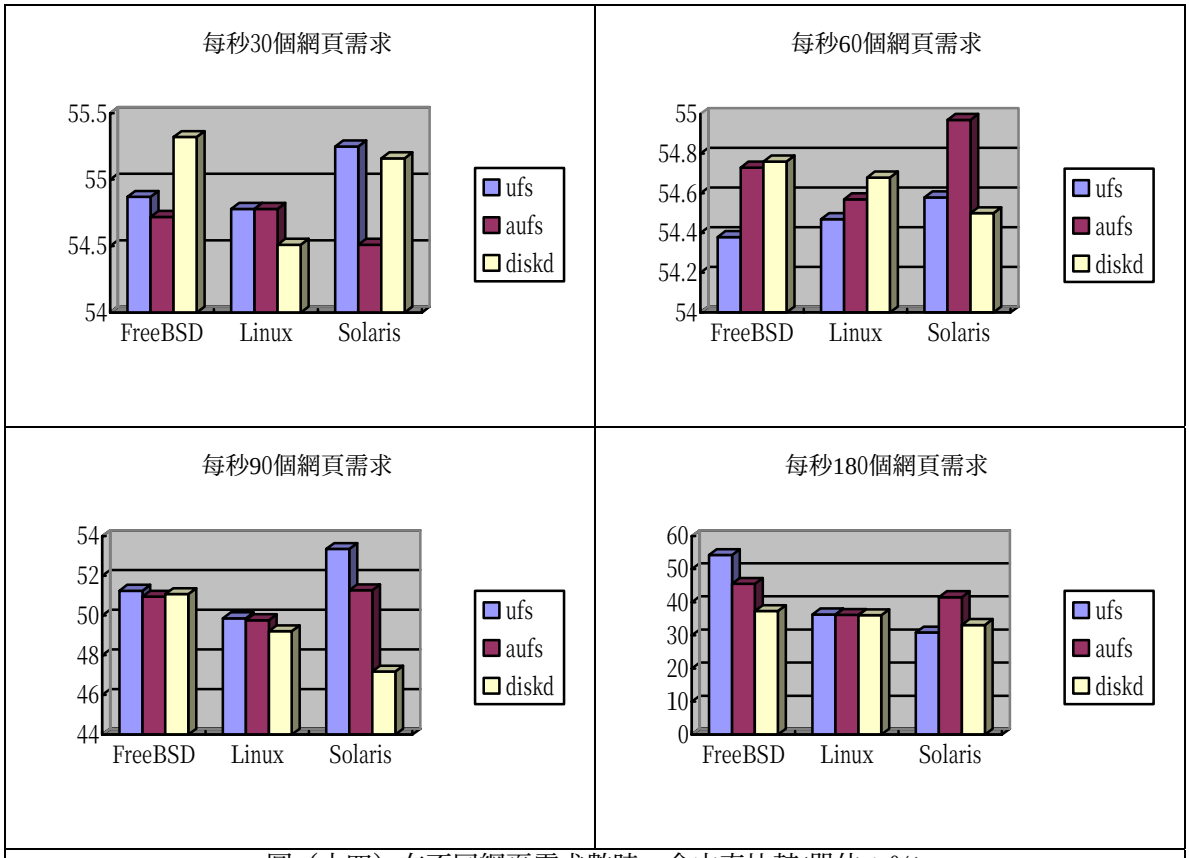
圖（十一）在不同網頁需求數時，擊中回應時間比較(單位：千分之一秒)



圖（十二）不同網頁需求數時，未擊中回應時間比較(單位：千分之一秒)



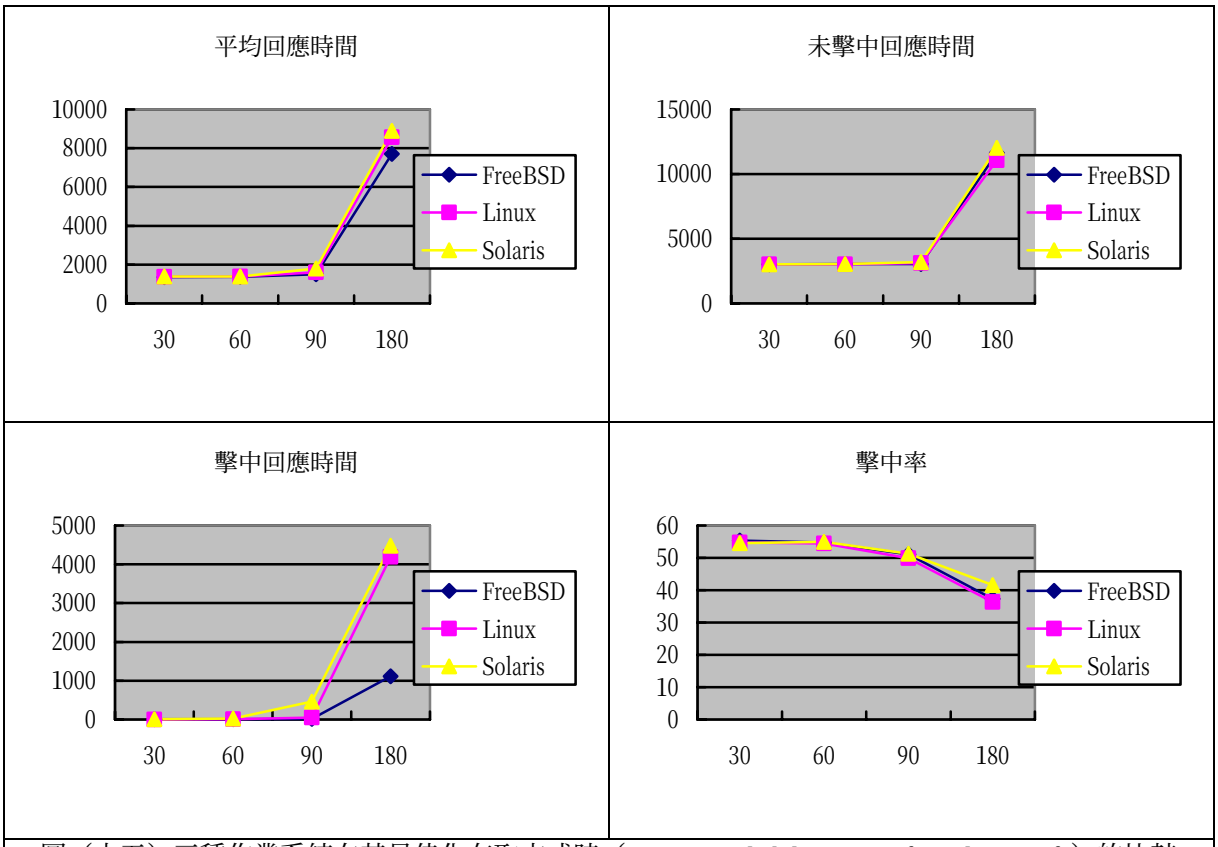
圖（十三）在不同網頁需求數時，平均回應時間比較(單位：千分之一秒)



圖（十四）在不同網頁需求數時，命中率比較(單位：%)

在圖（十四）的命中率方面，在需求數小於 60 時，皆有 54%以上的水準，表現很相近，但到了每秒需求數 90、180 時，就出現比較混亂的局面，Linux 表現依舊較為平穩，比較奇怪的是在 FreeBSD+ufs 每秒需求數 180 時回應時間極差，但卻出現 54.28%的高水準。

為了更清楚看出三種作業系統的差異，圖（十五）特別選擇每種作業系統最佳的存取方式（分別為 FreeBSD+diskd, Linux+ufs, Solaris+aufs）加以比較，發現低需求量結果是很相似的。



圖（十五）三種作業系統在其最佳化存取方式時（FreeBSD+diskd, Linux+ufs, Solaris+aufs）的比較

5. 結論

我們在本篇文章一開始介紹了一些 cache 產品及評比項目，發現了記憶體容量及網路卡存取是重要配備，結果在我們進行最後的測試過程，果然也觀察出這個情形，我們發現 FreeBSD 的表現較好是因為其幾乎沒有用到在硬碟的 swap 空間，善用記憶體儲存快取物件，當然會獲得較好的反應結果，反之，Solaris 則用了很多 swap 空間所以表現較差；此外，在進行較高需求數測試時，IBM PC Server 內建網路卡發生了很嚴重的封包遺失情形，FreeBSD 出現大量的”no memory for rx list – packet dropped!”，Linux 也出現大量的” TCP: drop open request from 140.126.22.117/36525,NET: 1065 messages suppressed.”訊息，但額外加裝了一塊 intel etherexpress

pro 網路卡之後，在 FreeBSD 的表現卻異常的好，使用 FreeBSD+diskd 甚至可以跑完每秒 570 個需求數，調高硬碟快取空間後，平均回應時間只有 6.4 秒內，是最好的選擇。

此外，在實驗測試中選擇的四種網頁需求，其實代表著四個等級的快取伺服器的流量，觀察了各區網中心的代理快取伺服器尖峰時期的使用紀錄，台灣大學[24]網頁上的四臺 proxy 平均每秒需求量在 50-80 之間、交通大學[25]三臺的主要 proxy 平均每秒需求量在 50 左右，中山大學[26]有四臺情形也跟交通大學相近，雖然這是每日的總平均值，但是一般每日的尖峰時間需求數，若扣掉平均睡眠時間八小時來算，平均是原來的 1.5 倍，即使尖峰使用量到了 10 倍，使用此次測試的平台安裝執行 squid，針對不同作業系統使用最佳的存取方式，還是可以符合需求的。只是 squid 的技術門檻較高，設定調整較不容易，但又因為其他快取產品的價格偏高，因此在學術機構比較能接受經濟又實惠的 squid 快取方案。

參考資料

[1] Squid Internet Object Cache: <http://www.squid-cache.org>

[2] The FreeBSD Project: <http://www.freebsd.org>

[3] Linux:<http://www.linux.org>, Linux.com – A Means to World Liberation<http://www.linux.com>

[4] Solaris product line: <http://www.sun.com/software/solaris/>

[5] Web Caching-related Papers and Articles: <http://www.web-cache.com/Writings/papers.html>

[6] Web Caching Products and Software: <http://www.web-cache.com/products.html>

[7] D. Wessels and K. Claffy National Laboratory for Applied Network Research/UCSD, Internet Cache Protocol version 2, RFC 2186, September 1997

[8] P. Vixie(ISC), D. Wessels(NLANR), Hyper Text Caching Protocol, RFC 2756, January 2000

[9] CARP, Cache Array Routing *Protocol***.** Internet Draft draft-vinod-carp-v1-03.txt(expired)

[10] WCCP, Web Cache Coordination Protocol. draft-wilson-wrec-wccp-v2-00.txt(expired)

[11] Ara Network Technologies: <http://www.aranetwork.com/eng/products/web.htm>

[12] CacheFlow cIQ Content Delivery: <http://www.cacheflow.com/products/index.cfm/>

[13] Cisco Cache Engine Series: <http://www.cisco.com/warp/public/cc/pd/cxsr/500/>

[14] Compaq TaskSmart C-series: <http://www.compaq.com/tasksmart/c4000/index.html>

[15] TMF(The Measurement Factory INC.): <http://www.measurement-factory.com>

[16] A. Rousskov and D. Wessels, The Third Cache-off. Raw data and independent analysis at <http://www.measurement-factory.com/results/>.

[17] ARPA(Advanced Research Projects Agency): Now called Defense Advanced Research Projects Agency (DARPA), the U.S. government agency that funded the ARPANET

[18] Harvest Project: <http://harvest.cs.colorado.edu/>

[19] 參閱下列網站：臺灣大學網路快取服務<http://procy.ntu.edu.tw>、中央大學網路代理伺服器<http://proxy.ncu.edu.tw>、交大網路快取服務<http://proxy.nctu.edu.tw>、新竹師院快取伺服器<http://proxy.nhctc.edu.tw>、中華大學網路快取服務<http://proxy.chu.edu.tw>、中正大學網路快取服務<http://proxy.ccu.edu.tw>、成功大學網路快取服務< http://proxy.ncku.edu.tw/>、中山大學代理伺服器< http://proxy.nsysu.edu.tw>、、、等

[20] The CodeRed Worm: CERT<http://www.cert.org> Incident Note IN-2001-09,13,19,23

[21] Squid Programmers Guide, Chapter 2 Overview of Squid Components, <http://www.squid-cache.org/Doc/Prog-Guide/prog-guide-2.html>

[22]] Squid Programmers Guide, Chapter 4 Flow of a Typical Request, <http://www.squid-cache.org/Doc/Prog-Guide/prog-guide-4.html>

[23] Web PolyGraph: <http://www.web-polygraph.org/>

[24] 臺灣大學網路快取服務 系統配備及使用統計: <http://proxy.ntu.edu.tw/ntu-proxy/stats.html>

[25] 交大網路快取服務 伺服器流量統計: <http://proxy.nctu.edu.tw/proxy/log.html>

[26] 國立中山大學代理伺服器首頁: <http://proxy.nsysu.edu.tw>