

## 一個支援文件分類之可擴展性代理伺服器

文成康、蔡尚榮、宋豐旭

成大電機系

台南市大學路一號

(Email: ckwen@eembox.ee.ncku.edu.tw)

### 摘要

網際網路技術快速的發展使得全球資訊網(WWW)之應用更為廣泛。網路上的大量網頁資訊讓使用者必須透過全球資訊網的代理伺服器(Proxy Server)或是搜尋引擎(Search Engine)才能較為快捷地擷取網頁資訊。然而過多的資訊也使得使用者對資訊的搜尋、過濾與分類耗時費力。因此，本文以網域名稱伺服器(DNS)結合代理伺服器及搜尋引擎為基礎，提出一個具簡易分類功能的可擴展性代理伺服器系統，以提高資訊擷取效率。

我們所實現的可擴展性代理伺服器系統具有支援文件分類的能力。伺服器群組可依負載輕重適度調整伺服器個數。本系統中的代理伺服器更可提供特定類別的網頁資訊，使用者可從中很快地找到興趣相近的資訊，大幅提升擷取效率，縮短回應時間(Response Time)，以及節省網路頻寬的使用。

關鍵字：可擴展性(Scalability)、代理伺服器(Proxy Server)、搜尋引擎(Search Engine)、文件分類(Document Categorization)、全球資訊網(World-Wide Web)。

### 1.0、簡介

隨著網際網路技術的快速進步，網際網路應用日漸普及，網路使用人口的急速增加，許多網路伺服器已不堪負荷。增加單一伺服器的硬體效能也趕不上使用者服務要求增加的速度，不但花費高，而且不符合經濟效益。因此必須設計多伺服器集合的伺服器群組(Server Cluster)來分散單一伺服器的負載，以低花費應付高需求。如何設計一個有效而實用的可擴展性伺服器系統，便成為在當前的 Internet 環境急需討論解決的問題。

由於全球資訊網(World Wide Web)的蓬勃發展，網路上的資訊越來越多圖形及多媒體的資料，使資料傳輸量相對增多，造成時間延遲。因此需要採用快取(Cache)技術來減少網路傳輸的延遲及 Web 伺服器的負荷，也就是代理伺服器(Proxy)的設計。可是資訊大量增加，也使得要在 WWW 中尋找所需要的資訊有如大海撈針，越來越顯困難，不但浪費了使用者的時間和精神，同時也造成了網路頻寬的浪費以及增加資訊伺服器的負載。因此如何讓使用者快速而有效地得到正確有用的資訊是目前所面臨必須解決的一個大難題。

為符合這個需求，我們便考慮將 Web 的資訊作分類，利用代理伺服器(Proxy)的設計，自行設計一具特定用途的代理伺服器，專門收集特定的某類資訊，以後當需要參考這些我們有興趣類別的資料時，透過這種代理伺服器，將會有很高的命中機率，因為這類文件已存在快取儲存體中了，如此可大量節省網路頻寬及使用者等待時間。但當全世界對這類資訊有興趣的人都透過這個代理伺服器來存取 Web 資料時，這個代理伺服器便會成為整個系統的瓶頸，而且距離此代理伺服器相當遠的使用者都來使用時，不但無法達到當初設計的優點，反而可能會更增加網路傳輸量及時間的浪費，於是這時便需要具備可擴展性的代理伺服器來解決這種問題。

本文的目標是在探討何謂一個具可擴展性的服務系統(Scalable Service)，及其隨之產生的許多問題，如何使網際網路服務具備可擴展性，並根據分析研究後提出一種可擴展性的系統架構，設計一個適用於廣域網路(WAN)的可擴展性伺服器群組，此架構的重心在於研究以現有的名稱伺服器(Domain Name Server)為基礎，搭配各種應用伺服器的運作，將數台伺服器聯合成一具有單一伺服

器影像的伺服器群組，達到負載分散的作用，縮短系統回應時間，提高服務的品質，並提供伺服器之間相當程度的容錯能力。而且不論對使用者或伺服器而言，這些設計都是透明的(Transparent)。伺服器群組的成員數目可隨著負載能力及使用者數目動態增減。此設計還具備一最重要的特點是與應用程式無關，現今在網際網路上的應用程式，只要是以用戶端及伺服器端(Client-Server)模式運作，均可在此架構下正常運作，不需做任何修改。

在目前的 WWW 環境中，Web 的文件是沒有任何分類機制可言的，而使用者要瀏覽 Web 網頁時，通常會透過代理伺服器(Proxy)來加快存取速度，但目前的 Proxy 的命中率(Hit Ratio)並不算高，越高的命中率才越能顯現出 Proxy 的好處，因此本文的另一個目標即是設計一支援文件分類的代理伺服器，利用 Proxy 蒐集不同類別的資料，當使用者透過此類 Proxy 去存取這類別的文件時，會有極高的機率是此文件已經存在 Proxy 的快取中，再搭配前述的可擴展性伺服器群組的設計，可使得這類 Proxy 具有可擴展性的能力。

我們將對於設計支援文件分類的可擴展性代理伺服器所面對的相關背景知識及論文研究設計與發展的過程，做一介紹。第二章探討一些 Scalability、Proxy、搜尋引擎及文件分類的相關知識，第三章敘述現有的相關研究，第四章說明利用 DNS 實現可擴展性伺服器群組的設計構想及實作細節，第五章說明如何設計一支援文件分類的代理伺服器，並應用可擴展性的設計，使其具有可擴展性的能力，第六章實際測試修改過的 DNS，其在現實應用的效能評估，第七章則為結論及對未來持續發展的計畫。

## 2.0、構想與背景知識

我們設計此系統是要提供一個具特定用途的代理伺服器，使其能收集特定類別的文件，透過利用不同的代理伺服器收集不同的類別，達到將 Web 文件分類的功用，使得使用者透過這類的代理伺服器去存取特定類別的資料時，大幅提高在 Cache 中的 Hit Ratio，節省網路頻寬及使用者等待的時間。而當系統規模變大後，便需要具

備可擴展性的能力，系統概觀如 Figure 1 所示。

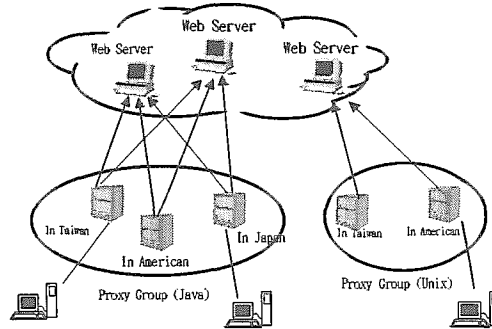


Figure 1 系統概觀

### 2.1 Scalable Service

為何 Scalability 已成為一項重要的問題，其最主要因素有：

1. 使用者數量的快速成長：大量的使用者要求會造成網路交通的擁擠及伺服器負載的急速增加。
2. 網路上資料物件的大小增加：多媒體的世界，這些多媒體資訊的大小是傳統文字檔的千萬倍，往往造成了網路和伺服器容量的瓶頸。
3. 網路上資料量的大量增加：資料量的增加使得存取這些資料，資料搜尋，資料管理更為困難，同時也增加了系統負載及處理時間。
4. 使用者需求的分佈不規則：由於使用者的位置、時間及習慣不一，伺服器的使用可能很不平均，造成負載不平衡的情形。

要如何建構一可擴展性的伺服器，提供一具可擴展性的網路服務呢？傳統可行的方法有以下幾種：

1. 快取(Caching)：利用 Cache 或 Proxy，讓使用者可以就近取用資料，可以節省存取時間，減輕網路和伺服器的負載，增加可利用性。
2. 複製(Replication)：將資料或伺服器複製多份，分散在不同地方，可服務不同區域的使用者。
3. 分割(Partition)：將資料或整個系統服務分割成許多部分，由不同的伺服器負責，如此整個系統的負載便可分散在不同的伺服器上。
4. 動態分配使用者要求(Dynamic request mapping)：可以根據伺服器位置、客戶端位置、伺服器負載及網路狀況，動態的將使用者的要求分配到不同的伺服器上。

目前採用的可擴展性設計應用在伺服器群組有以下幾種方法：

### 1. HTTP-redirectation 的設計

在 HTTP 的協定中，當一個 Web 伺服器負載太重時，它可以對客戶端發出的要求回應 302 的訊息代碼，並加上另一個 IP 位址，當客戶端收到這個訊息代碼及 IP 位址後，便會重新發出另一個要求到新的位址，這樣便可以把負載分散到其他的伺服器上。

### 2. 客戶端的設計

在客戶端要達到可擴展性的設計目前有兩種方法，第一種是讓使用者知道伺服器群組的分配，由使用者自行更改伺服器位址，使得使用的發出的要求分散到不同的成員伺服器中，達到分散負載的效果。第二種方法是採用美國 Berkeley 大學發展的 Smart Client [3]，它是利用在客戶端執行的一個 Java Applet，去收集所有伺服器群組的成員伺服器上的負載程度和一些其他相關資訊，再決定把客戶端的要求分配到其中一個成員伺服器上。

### 3. 伺服器端利用名稱伺服器(DNS)的設計

在伺服器端可利用名稱伺服器[7][8]IP 位址的對應時，將一個 Domain Name 對應多個 IP 位址，名稱伺服器回應客戶端發出的名稱與 IP 位址的轉換要求時，會將 Domain Name 輪流對應到 IP 列表中的一個，如此便會將客戶端的要求分散到各個成員伺服器上。

### 4. 伺服器端利用單一 IP 位址影像的設計

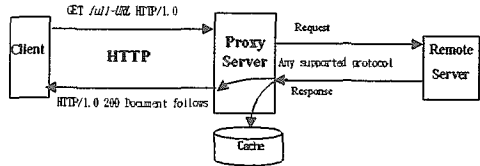
在 Internet 上要提供單一的伺服器影像，一般可從兩方面著手：一種是提供相同的 Domain Name，另一種是提供相同的 IP address[2]。

## 2.2 Proxy

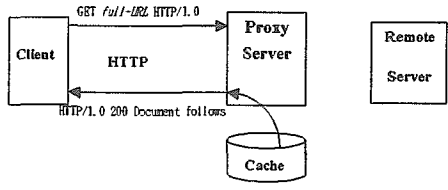
對於同一份熱門文件的擷取，不論距離遠近都得到原始伺服器去取得的話，勢必造成伺服器的負載大增，而且遠距離的使用者可能必須經過很長的網路傳輸才能得到所需的文件，對網路更造成了相當大的傳輸量，因此如果在各個區域都有相同伺服器的複製，不但可以減輕來源伺服器的負荷，更可以大幅減少網路上的交通量，縮短使用者等待網路傳輸的時間。但是 Web 的資訊並不適合完全複製(Mirror)，所以採用另一種方法來解決前述伺

伺服器負荷過重及網路延遲和頻寬浪費的問題—快取(Cache)，而 Proxy 即是用快取的方法來解決以上問題。

Figure 2 是 Proxy 的基本運作方式：



(A) Cache Miss



(B) Cache Hit

Figure 2 Proxy 的運作圖

## 2.3 文件分類

在 Internet 的環境中，Usenet 的新聞討論區(News)是被使用者廣泛使用的，新聞討論區是以主題為依據，分成眾多的討論群組(Newsgroup)。但在現今的 Internet 環境，Web 的資訊並沒有一個像新聞討論區一樣完整的分類機制，如果使用者想要瀏覽某類主題相關的網頁，並沒有辦法像新聞討論區一樣。因此，使用者想要真正找到想要瀏覽的網頁，可說是相當困難的。目前已提出可對 Web 網頁做分類的構想及設計有 Meta Tag[14]和 PICS[11][13]。但是這樣作法，每次存取一個新的文件都要取得分類規則再加以判斷，效率不好，而且需要所有的網頁提供者配合，遵循相同的規則去做，在實行上有較高的困難性。

## 3.0、相關研究

NCSA 的 Web Server [1]為了應付越來越多的使用者，利用 DNS 內含的 Round-Robin 功能，設定數部不同 IP Address 的伺服器使用相同的一個網域名稱

(www.ncsa.uiuc.edu)。當使用者發出的要求送往 www.ncsa.uiuc.edu 之前，會先向 DNS 查詢其 IP Address，此時 DNS 會循序以 Round-Robin 方式將其對應至不同的伺服器，而達到負載分散的效果。但因為 DNS 本身的階層式及快取的設計，使得負載分散的效果並不好。

Cisco 公司所發展的 DistributedDirector [12]是用來解決廣域性的可擴展性問題，它提供廣域的可擴展性能力，增加了伺服器的效率，並減少網路使用的頻寬浪費。

PICS-aware Proxy Server [11]的架構主要包括兩個部分：一個是標準的 Proxy Server 模組，另一個是 PICS 處理模組。在此，Proxy 只是扮演了類似防火牆的角色，來過濾資訊，無法完全對所 Cache 的資訊加以控制。

#### 4.0、系統設計與實作

##### 4.1 設計考量

我們修改 DNS 的設計，來達到廣域可擴展性伺服器的設計，並且解決採用 Round-Robin 所會遇到的問題，利用 DNS 所提供之網域查詢功能，以及 PING 的回應功能，設計出一套可供伺服器群組使用的系統，透過此系統，用戶以伺服器的網域名稱便可使其應用程式連接至較近且回應速度較快的伺服器，此系統亦可使伺服器群組之設計，不須侷限在同一網路中，可分散至世界各地，達到真正具有可擴展性之伺服器架構。

##### 4.2 系統架構及運作模式

本系統可由 Figure 3 說明，圖中之 WWW 網路服務是由三部獨立的 Web 伺服器形成一個伺服器群組來提供相同的服務，此服務具有單一網址：www.ncu.edu.tw，各伺服器亦有獨自的 IP 位址，當用戶欲以此服務網址取得 WWW 服務時，瀏覽器必須查詢當地的 DNS 取得對方的 IP 位址，以便連線，當 DNS 發現此 Domain Name 對應多個 IP 位址時，此時，DNS 便會針對三個伺服器送出 ICMP 的訊息，並等待回應，然後 DNS 再把第一個回應的伺服器位址，調整為 IP 列表的第一個，再將此 IP 列表回覆給用戶，用戶端的瀏覽器便依此位址進行連

線，取得所要的資訊。

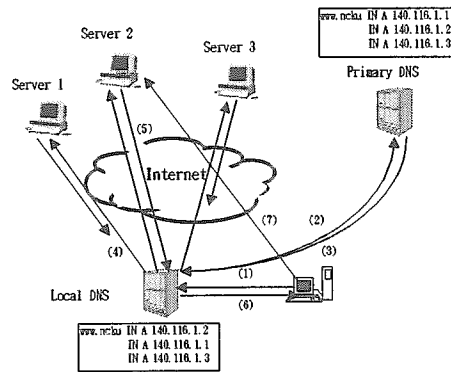


Figure 3 提供可擴展性伺服器群組的 DNS 設計運作流程

由於此三個伺服器除了可以在同一區域網路中，亦可分散在不同的網路，而且又因它們皆可提供相同之服務，因此，用戶可就近連接伺服器，亦可避免某一伺服器故障時，所造成的服務停止之現象，而伺服器的個數也可視各伺服器負載之輕重，適當地增加或減少。

本系統之設計重點在於使 DNS 與 PING 功能相結合，一般而言，當用戶以網域名稱對當地的網域名稱伺服器進行查詢時，此伺服器會先檢視本身的資料庫以取得名稱所對應的網路位址，若此網域名稱不屬本地的話，它會再至遠地的主要 DNS 伺服器查詢，然後將查詢到的結果回應給用戶，然而，若是此網域名稱所對應的網路位址不只一個時，則回應給用戶的將是一連串位址，但用戶端將只取用第一個位址，雖然可以利用 Round-Robin 的方式，使用戶可以依序取用到此一連串位址中的不同位址，但是此法所得到的位址，無法保證所對應的伺服器目前是否正常運作中，而且也無法知悉本地至目的地的伺服器的網路連線是否順暢，以上種種問題，使我們認為 Round-Robin 的方法並不適用在設計具擴展性的網路伺服器上。

為了在眾多的伺服器位址中，挑選出正常運作中的伺服器，而且要距用戶端近或是網路連線速度較快的伺服器，我們利用網際網路的 ICMP 協定來達成，當 DNS 依據用戶提出的伺服器網域名稱查詢，將得到一連串伺服器群組中各伺服器的 IP 位址，DNS 再針對這些位址，

個別送出 ICMP 訊息,然後進行等待回應之訊息,當 DNS 接到第一個回應訊息時,便可決定調整在 IP 位址列表的順序,將首先回應的伺服器位址調整為一連串位址的第一個,再將這一連串位址回應給用戶,用戶再依此位址進行連線工作,若是 DNS 經過一定時間的等待後,無法接到任何一個伺服器的回應,則不對原來的 IP 位址列表的順序作調整,直接將原來得到的一連串位址,不做任何更動,回應給使用者,這樣可減少使用者等待 ICMP 回應的時間,利用這樣的作法,回應最快的伺服器,代表它和用戶之間的網路較為順暢或是距離較近,這正是我們期望用戶可連接上的伺服器,反之,沒有回應或是回應時間過久的伺服器,則可能已故障或是網路的延遲嚴重,這當然不是我們所要選擇的。

由於 DNS 的運作過程中,會藉助快取(Cache)來暫存位址資訊,以提升查詢效率,但這會影響到當伺服器群組中伺服器個數有所增減時,無法及時反應此變化,此一問題可在主要的 DNS 上將此伺服器群組項目的 TTL(time-to-live)值調低即可應付。

在本系統的設計下,伺服器群組的組態異動上的管理是很容易的,當有需要新增或減少伺服器群組中的伺服器成員時,例如,原有的各伺服器負載均已在高檔,需加入伺服器分擔負載,或是伺服器使用率偏低,可移除部份伺服器,此時,伺服器群組的管理人員只須重新向主要的 DNS 登錄新增伺服器的位址或移除不用的伺服器位址即可。

### 4.3 系統特色及優點

我們所設計的這個系統可具有以下幾個特性：

#### 4.3.1.可擴展性之設計

當服務隨著時間的成長,而使用者逐漸增多時,單一伺服器已無法滿足使用者的需求,此時便必須以伺服器群組來代替單一伺服器提供服務,而伺服器群組的伺服器成員個數可能根據負載隨時有所更動,經由此系統的設計,服務提供者可以很方便的動態加入或減少其伺服器群組的成員個數,只需簡單的在主要 DNS 伺服器上的資料庫增加或減少對應的 IP 位址個數即可,最多經過一個

TTL 的時間,所做的更動便會反應在各個使用者的查詢,而伺服器成員的位址可分佈在各處,不拘限於區域網路(LAN)中,可真正達到可擴展性的目標。

#### 4.3.2.單一伺服器外觀

如果提供服務的伺服器群組能擁有單一的伺服器外觀(Single Server Image),對使用者來說這樣的系統可以說是完全透明的(Transparent),不論是單一的伺服器或是有多個伺服器成員的伺服器群組,使用者只要利用相同方式,對同一個伺服器影像發出服務請求即可。

本系統利用 DNS 伺服器將單一 Domain Name 對應到多個 IP 位址的功能來設計可擴展性伺服器,因此不論單一伺服器或伺服器群組,使用者只看到唯一的 Domain Name,提供了單一的伺服器外觀。

#### 4.3.3 良好的負載分散能力

傳統的 DNS 採用 Round-Robin 的方式,只是達到負載分配,並沒有真正做到負載平衡,而且並沒有考慮到網路實際狀態,尤其當伺服器分佈在各地時,並無法考慮到實際網路距離的因素,本系統的設計與傳統的 DNS 不同,當一個 Domain Name 對應多個 IP 位址時,可利用送出 ICMP 訊息找出網路速度最快的伺服器,使得使用者可以找出最近最好的伺服器,當伺服器的負載變高時,相對的回應 ICMP 訊息的速度也會變慢,因此分配使用者的要求到此伺服器的機會也會變低,可做到根據伺服器的負載能力及當時的網路狀況來分配使用者的要求。

#### 4.3.4 極佳的可利用性

本系統的設計提供了極高的可利用性,當伺服器群組中有某些機器當掉或網路不通而無法提供服務時,此機器也不會回應 ICMP 的訊息,因此使用者得到的結果一定是可提供服務的機器,使用者根本無法發覺其中哪些伺服器因發生錯誤而無法提供服務,因此此種作法也達到了容錯(Fault Tolerance)及錯誤透明性(Fault Transparent)的能力。

#### 4.3.5 與應用無關性

本系統只需修改區域的 DNS 部分，所有客戶端及伺服器端的應用程式均不需有所更動，因此現今所有 Internet 上的應用程式均可正常運作，不需做任何修改，對服務提供者及使用者來說，都是非常方便的。

### 5.0、應用實例—支援文件分類的可擴展性代理伺服器

#### 5.1 構想

如果把同類的資料收集在 Proxy 中，使用者可以透過附近的 Proxy Server 去存取此類資料時，命中率(Hit Ratio)會很高，可以節省許多的網路傳輸時間，而且不同類的資訊放在不同的 Proxy Server，可略微達到資訊分類的功能。

這種設計在使用這個 Proxy 的使用者有相近的喜好，對於某類資訊特別有興趣時，更能顯現出這個 Proxy 設計的好處，如：同一個實驗室的研究群、同一個公司的員工...等，由於使用這個 Proxy 的使用者所存取的資料大多是相似的類別，因此此類別的資訊在 Proxy 的 Cache 中的 Hit Ratio 相對會變的更高，透過這個特殊用途的 Proxy，便可以節省這些網路頻寬和使用時間。

利用 Proxy 的設計還有一個好處便是著作權及版權的問題，如果單純的利用一個 Web Server 來收集特定類別的資料時，便會出現前述 Mirror 一個 Web Server 所會遭遇到的不便，不但失去了原有的彈性，還可能有違反著作權及版權的疑慮，而在目前，使用 Proxy 來 Cache 資料是合法的，這也是我們採用 Proxy 設計的另一個重要原因。

將此系統與前述的可擴展性伺服器設計結合，蒐集同類相關資訊的 Proxy Server 不只一個，可能分佈在世界各地，我們可以把蒐集同類資訊的 Proxy Server 當作一個伺服器群組，有唯一相同清楚易記的 Domain Name，例如：java.proxy.com，book.proxy.com...等，使用者不須知道蒐集某類資訊的 Proxy Server 的真正位址為何，當要尋找有關 java 的資訊時，只需將其瀏覽器的 Proxy 設

定為 java.proxy.com，透過之前所提出的 DNS 設計，便可自動連到離使用者最近的蒐集 java 資訊的 Proxy Server，對使用者來說是完全 Transparent 的。

#### 5.2 設計目標

我們設計此支援文件分類的可擴展性代理伺服器，期望可達到下列目標：

1. 蒐集某一特別領域的文件。
2. 讓管理者透過 WWW 網頁便能管理此系統。
3. 用 Proxy 加上搜尋引擎，以期能達到簡單的資訊分類功能。
4. 將相關分類的資訊都 Prefetch 到 Proxy 中，可大量節省網路傳輸及等待的時間。
5. 可將某些 URL 由 Proxy 在網路較空閒的時間幫我們把所要的網頁先做 Prefetch 的動作，節省使用者的時間。

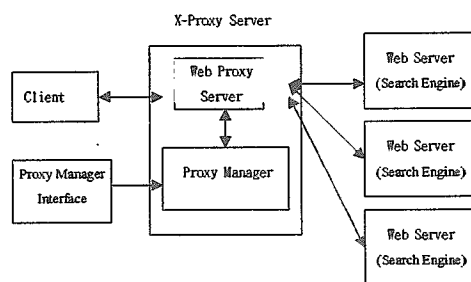


Figure 4 支援文件分類之代理伺服器系統架構圖

#### 5.3 系統架構

Figure 4 是本系統的架構圖，茲對主要部分做一介紹：

Proxy Manager Interface：是 Java Applet，提供圖形化的管理介面，讓管理者可以透過瀏覽器從遠端電腦直接設定一些參數，可控制 Proxy Manager 的執行動作。

Proxy Manager：是由 Java 實作的一個 Daemon Application，利用 Socket 和 Proxy Manager Interface 的 Applet 通訊，接收管理者傳來的參數及設定，再按照管理者的設定，透過 Proxy 向不同的 Search Engine 或 Web Server 發出請求，因此這些要求所得到的文件資料便會存在 Proxy 的 Cache 中了。Figure 5 為本系統運作流程。

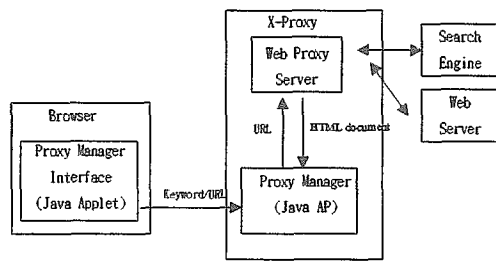


Figure 5 系統運作圖

#### 5.4 系統特色

我們所設計支援文件分類的可擴展性代理伺服器具有以下特性：

1. 利用搜尋引擎的查詢蒐集特定類別的資訊，不同的 Proxy 蒐集不同類別的資料，管理者也可指定相關此類的網站，先利用 Proxy 去 Cache 這些資料。
2. 可指定特定的 URL，透過 Proxy 在網路負荷比較輕的時候，去做 Prefetch 的動作。可以避免因網路負荷過重而增加使用者等待網路傳輸的時間。
3. 連線過程發生錯誤，可定時重新嘗試連線的功能。
4. 採用多執行緒(Multithread)同時向多個搜尋引擎查詢，可大幅增加效率。
5. 利用 Java Applet 提供一友善的管理介面，只需用瀏覽器，便可在遠端管理 Proxy Manager 的運作。
6. 配合之前的 DNS 設計，此支援文件分類的代理伺服器也具備了可擴展性。

#### 6.0、系統效能測試

我們對修改後的 DNS 做效能測試，以了解此 DNS 設計在 Scalable 方面真正達到的效果及可得到的好處，並瞭解我們對 DNS 的修改對效能上造成多大的影響，作為未來所力求改進的地方。

測試方法：Client 端以 Ping 當作測試用之應用程式，對每個 Domain Name 發出查詢，每次送出 50 個 packets，這樣可同時看出 Ping 的結果，及修改過的 DNS 所得到的改善。

測試環境：DNS Server：AMD5x86-133，20MB RAM，10Mbps Ethernet。DNS 版本：BIND 4.9.6[6]。作業系統：FreeBSD 2.2.5。實驗組為修改過之 DNS，對照組為原來

#### 未經修改之 DNS

測試分別為在 LAN、LAN+WAN、WAN 上用 Ping 模擬一般應用程式的網路連線狀況，結果可看出我們所設計的 DNS，除了在應用程式執行剛開始時，因為要做 DNS 查詢，需要花費較多時間外，之後的連線時間均相當接近速度最快的 Host，更比使用原來 DNS 連線的平均時間減少許多，在 LAN 上的測試相差不大，但在 WAN 上的測試便可很明顯的看出效率上的改善，而且應用程式執行的時間越長，效率上的改善越為明顯。

此外，修改過的 DNS，在查詢上要花費較多的時間在執行新加入的 Ping 功能函式，所造成的 Overhead 並不算小，但經由此修改過的 DNS 去執行 Client 的應用程式時，由數據可看出可大幅減少網路傳輸及使用者等待的時間，而且 Packets 的 Loss 也變得很小，可增加網路傳輸的正確性及穩定性，因此在 DNS 部分多花費一點時間可得到更順暢的網路傳輸，可說是相當值得的。

#### 7.0、結論與未來展望

在本文中，我們討論了 Scalability 問題的原因，及增加 Scalability 能力的各種相關研究，並且透過修改 DNS 的設計來實現在廣域網路的可擴展性伺服器群組，能夠提供單一伺服器影像，達到負載分散的作用，改善回應時間，而且提供伺服器之間的容錯能力，伺服器成員的數目可以隨時動態的增加或減少，還具有與應用程式無關的特性，可使得現有的 Internet 應用程式不做任何修改而均可正常運作。

透過我們所設計可支援文件分類的 Proxy，利用搜尋引擎來達到蒐集不同類別資料的目的，提高 Hit Ratio，大量減少網路頻寬的浪費及使用者等待的時間。此系統還提供了針對指定網站的 URL，做 Prefetch 的動作，可將某些 URL 由 Proxy 在網路較空閒的時間幫我們把所要的網頁先做 Prefetch 的動作，或者管理者可將此 Proxy 所收集的類別中較有名的或使用者較常去的網站，先做 Prefetch 的動作，或定期更新，如此對於使用者瀏覽此類的資訊時，可以有很大的幫助。

將我們所設計的 Proxy 和提供可擴展性伺服器群組的

DNS 設計結合，便可成為一完整支援文件分類的可擴展性代理伺服器。將分散於各地蒐集同類資料的 Proxy 合成一代理伺服器群組，使用者透過本系統中的 DNS 可找到離自己最近的 Proxy，而同一代理伺服器群組中的各個 Proxy，可互為 Parent 或 Sibling 的階層式架構，可提供更有效率的運作。

#### 參考文獻

- [1] E.D.Katz, M.Butler, RMcGrath, "A Scalable HTTP Server: The NCSA Prototype", Computers networks and ISDN systems, Vol 27, 1994, pp.155-164
- [2] O. P. Damani, P. Y. Chung, Y. Huang, C. Kintala, and, Y. M. Wang, "ONE-IP: Techniques for Hosting a Service on a Cluster of Machines," in Proc. The Sixth Int. World Wide Web Conference, April 1997.
- [3] Chad Yoshikawa, Brent Chun, Paul Eastham, Amin Vahdat, Thomas Anderson, and David Culler, "Using Smart Clients to Build Scalable Services", Computer Science Division University of California Berkeley, CA 94720
- [4] Antony Y. R. Lu, "A Scalable Server Broker Supporting Single Server Image and Load Balance", NCKU, 1997
- [5] "Extensible Markup Language(XML)", <http://www.w3.org/XML/>, 1998
- [6] Berkeley Internet Name Domain Server, BIND 4.9.6, <ftp://ftp.isc.org/isc/bind/src>, 1997
- [7] RFC 1034, "Domain Names - Concepts and Facilities"
- [8] RFC 1035, "Domain Names - Implementation and Specification"
- [9] E.Anderson, D.Patterson, and E.Brewer. "The MagicRouter, and Application of Fast Packet Interposing". May 1996.
- [10] <http://www.cisco.com/warp/public/cc/cisco/mkt/scal e/locald/index.shtml>.
- [11] Wayne B. Salamonsen, Roland Yeo, "PICS-Aware

Proxy System Versus Proxy Server Filters",[http://www.isoc.org/isoc/whatis/conferences/inet/97/proceedings/A7/A7\\_3.HTM](http://www.isoc.org/isoc/whatis/conferences/inet/97/proceedings/A7/A7_3.HTM), 1997

- [12] "Cisco Distributed Director", [http://www.cisco.com/warp/public/751/distdir/dd\\_wp.htm](http://www.cisco.com/warp/public/751/distdir/dd_wp.htm), 1998
- [13] "Platform for Internet Content Selection", <http://www.w3.org/PICS/>, 1997.
- [14] "A Dictionary of HTML META Tags", <http://vancouver-webpages.com/META/metatags.detail.html>, 1998