Secure Server Switching System with extensibility

具擴充性之交替式安全伺服器

王鈺勝
逢甲大學資訊工程學系
m9702772@fcu.edu.tw

王　壘
逢甲大學電機工程學系
leiwang@fcu.edu.tw

摘要－近年來因為網路服務應用的快速成長，隨之衍生而出的網路安全問題更是層出不窮：諸多伺服器成功入侵攻擊之資安事件提醒我們安全保護不該只著眼於入侵事件發生之前的防護措施，在面對攻擊者成功入侵系統後的容侵技術議題亦不容小覷。為此，本篇研究提出在各主機中利用虛擬機器之技術，達到動態的伺服器服務引擎的快速置換與入侵排除：讓系統遭遇入侵攻擊也能在極短時間內恢復正常，以盡可能減低損失。

本論文基於先前提出具備高擴充性之交替式安全伺服器架構，以避免原先中央控制器在未來可能成為阻礙系統擴展之瓶頸。在實作上以嵌入式系統技術開發中央控制器，以達到對整個伺服器群的協調與執行分配，使一群主機資源能滿足大量的伺服器系統的處理需求，在平衡分配伺服器的原則下充分利用所有的運算能力，同時利用冗置之概念，使系統的故障不會導致服務錯誤或中斷，達到容錯理論中優雅退化的目的。

Abstraction － Protective measures for server invasions should not solely focus on events before an invasion occur. Recording and monitoring successful server invasions with endless streams of security mechanisms should be employed attentively to reduce the loss of data due to successful intrusion attacks on any system.

Overlooking this will inevitably lead to detrimental data loss and stray away from the key issue of limiting the time an unauthorized user has on the system.

This paper presents an implementation of embedded system technology developed in a host control module group in which the entire server achieves coordinated group allocation of resources combined with a larger number of group hosts designed to meet demand. The proposed security server switching system uses server load balancing to prevent system failures, errors, and interruptions, accompanied with the ever so important theory of fault-tolerance for grace degradation purposes.

The prototype described in this paper is comprised of 8 servers with 2 hosts housing those servers to meet service demand .In addition, this VM to host relationship allows 4S to recover from attacks rapidly by allowing the improvised service on server the ability of being back to normal in a very short time.

關鍵詞－冗置技術、入侵容忍、入侵消除、虛擬機器

Keyword － Redundancy 、 Intrusion Tolerance 、 Intrusion Elimination、Virtual Machine

一、簡介

當今網際網路應用服務架構主要為主從式伺服器模式(Client-Server Model)－由提供服務者建構一伺服器並接受來自各方的客戶端服務要求，再將回應訊息傳回給客戶端，舉凡網頁服務、網域名稱服務、檔案傳輸服務等應用均係以此模式提供服務。

在社會文明的數位化與網路化驅使之下，促使今日蓬勃發展的網際網路服務世界，許多與民生、文化、教育與商業交易等活動幾乎無一不與網際網路掛?，如此普及的數位網路生活型態足以顯示伺服器系統對人類社會文明的重要性與不可取代性。

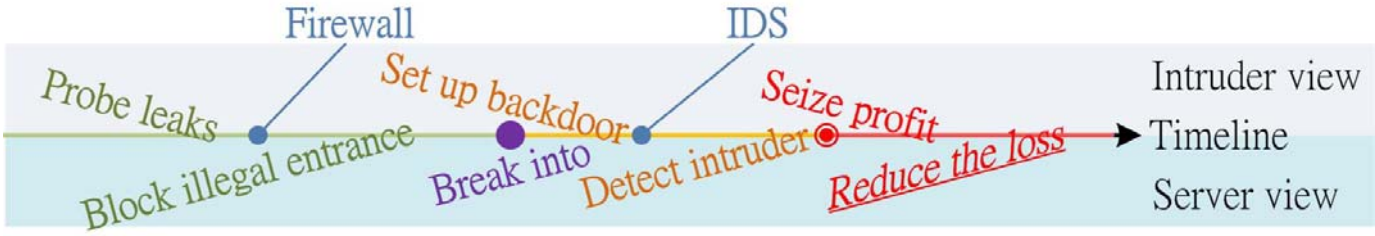

圖 1 入侵攻擊進程示意圖

試想，假若提供各式服務的伺服器系統因人為的惡意破壞或是發生非預期的系統故障，甚或是重要數據資料遭到毀壞或竊取等情況，其造成的嚴重性將可能癱瘓社會的運作機能，甚至走向世界大亂的局面。

雖然已發展出許多穩定成熟的各式安全防護架構保護伺服器系統，如：防火牆(Firewall)、入侵偵測系統(Intrusion Detection System)等防護措施；然而近年來頻傳的伺服器系統遭網路怪客(Cracker)成功入侵之資安事件層出不窮，讓我們體悟到欲建構猶如銅牆鐵壁的安全防護罩在資訊安全領域中是難以達到及確保的。

如圖 1，說明入侵者與系統提供者在攻擊進程中的關係。在過去許多安全研究技術著眼在如何建立一個穩固的安全伺服器環境將惡意攻擊者阻擋在系統之外，如防火牆即是最前線的保護，其主要功能就是要阻擋攻擊者的成功連線。為補足第一道防線的不足系統管理員往往會設置入侵偵測系統作為第二道防線，依類型可分為網路偵測入侵系統與主機系統入侵偵測系統，分別對封包訊息或是主機行為進行檢測，以期糾出成功穿越防火牆的入侵者。

根據 2007 年的賽門鐵克安全威脅報告[9]指出，龐大的地下經濟利益促使攻擊行為走向專業化與商業化，並且攻擊行為亦轉向複雜的多階段式攻擊手法：攻擊者首先成功侵入主機並且完成後門設置，待潛伏一段時間後才會接連展開獲取利益的攻擊行為，當系統管理員偵測到系統漏洞時，往往是在入侵者成功獲取利益之後，並且系統遭入侵造成的損失總是隨著入侵者的停留時間的增加而增加，因此有必要將入侵攻擊的進程縱深繼續往下延伸：如何降低防護機制所疏漏的攻擊事件造成的損失(金錢甚至是組織信譽的損失)。

有鑑於上述的考量，本研究提出名為「交替式安全伺服器」(Secure Server Switching System，簡稱為 4S)之系統架構以完成入侵容忍(Intrusion Tolerance)上的安全議題[11]。我們基於一較為悲觀但卻實際的觀點看待網路安全：沒有絕對安全的伺服器保護措施可以百分之百地成功阻擋入侵者的攻擊，因此交替式安全伺服器著眼於如何減少攻擊者成功入侵系統之後的駐足時間。

在交替式安全伺服器架構下包含一組伺服器群(Server Pool)與一個伺服器群控制裝置(Pool Controller)。伺服器群採用虛擬機器之技術[1][5][8][10]建構而成並且配合虛擬機器之快照功能(snapshot)使伺服器狀態能夠完成快速置換以達到入侵排除(Intrusion Elimination)之效果；本研究以嵌入式技術開發伺服器群控制裝置，其負責整個伺服器群的協調與執行分配，使數台主機(Host)之資源能滿足伺服器系統(Server)的處理需求，在平衡伺服器分配的原則之下充分發揮運算資源，並且使用冗置(Redundancy)之技術讓部分主機故障亦不會導致服務中斷或錯誤發生。

本論文架構說明如下：第二節介紹說明與本研究相關之研究技術；第三節針對本研究提出之具擴充性交替式安全伺服器架構作一全面性的介紹；第四節介紹完整的系統實作與實測數據比較；最後第五節為本研究之結論與探討。

二、相關研究

本節介紹與本研究所提出的 4S 系統相關的基礎研究，包括自我清除容侵機制(Self Cleansing Intrusion Tolerance，SCIT)與虛擬伺服器(Linux Virtual Server，LVS)技術。

(一) 自我淨化容侵系統(SCIT)

SCIT[3][14][15]的研究概念主要著眼於加深系統安全防護的縱深，保守的假設任何嚴密的安全防護均可能招致惡意的攻擊及入侵，因而絕對不能單只依靠某一或某些系統防護功能即自認安全。

SCIT 採取系統交替換手並自我清除重新開機 (Self-Cleansing) 的方式，使伺服器曝露於開放的網路環境之時間受到限縮。在 SCIT 的作業環境及 Cleansing 及 Reboot 的基本動作上，是利用虛擬機器(Virtual Machine)的技術，在單一的主機上建構多個相互隔絕的 Virtual Machine 環境，並透過 Central Controller 的軟體程式下達命令給 VMM(Virtual Machine Monitor)進行虛擬機器之管理以達到伺服器淨化之目的。

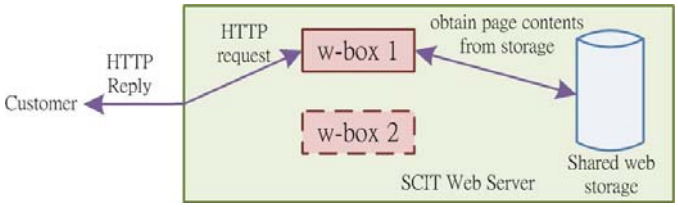

圖 2(a) SCIT 運作示意圖 1

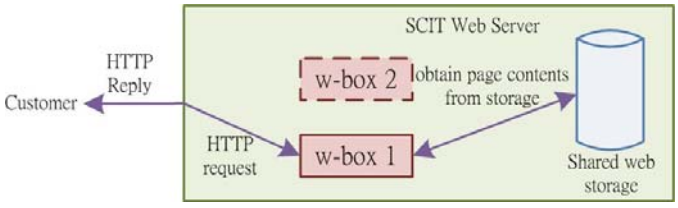

圖 2(a) SCIT 運作示意圖 2

如圖 2(a)與圖 2(b)所示，配合獨立的控制硬體主導切換、與外界隔絕的系統資料庫負責自我 Cleansing 及 Reboot，如此定期性的切換伺服器使入侵者即使入侵成功也無法在有限的時間內達到竊取、竄改資料的目的。

Anantha K. Bangalore 與 Arun K Sood 在 2009 年提出的文獻[2][7]中針對 SCIT 應用在 Web 伺服器上的測試結果證實採用虛擬機器技術建構伺服器所造成之負擔十分輕微。另外，運用馬可夫鏈推演出 SCIT 運作模型的可用性公式並證明縮短伺服器暴露的視窗時間將可提升系統之容侵能力。

SCIT 架構的主要缺陷在於缺乏延伸性(Scalibility)與可靠性(Relibility)的考量，該架構僅限於單一實體主機，在面對硬體設施的故障情形顯然缺乏應對能力，並且對於多變的服務需求及負載無法利用備份的處理能力以靈活支援伺服器的服務負載，無法在日後動態的擴充伺服器規模。

(二) 虛擬伺服器(LVS)

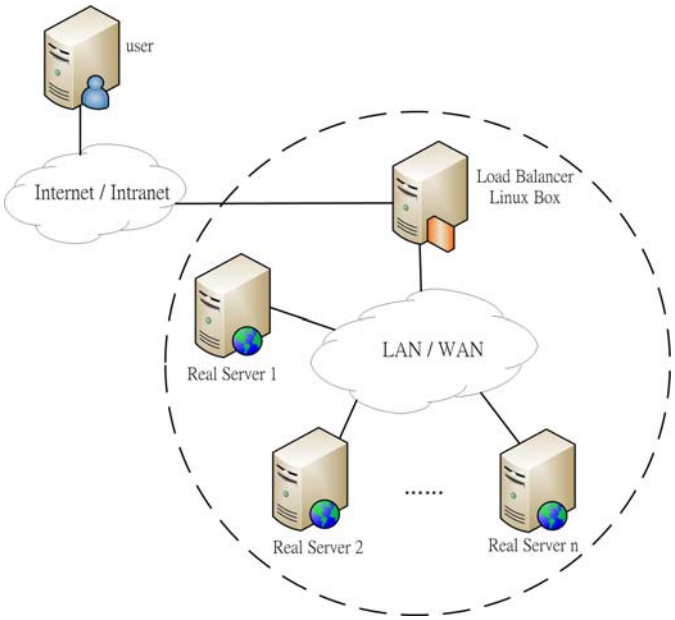

圖 3 Linux Virtual Server 架構示意圖

因應網際網路服務的重要性日益提升，越來越多重要的伺服器主機必須提供全時段不停機的可用性，並且要能夠承受突發性的巨量服務要求。虛擬伺服器(Linux Virtual Server，LVS)架構[12][13]即是為了達到如此高延展性(Scalability)

與高可用性(Availability)之目的而發展之技術。

LVS 系統架構參考如圖 3 所示，面對外部網際網路設置一 Load Balancer 裝置作為對外的單一溝通窗口，使用者看到的系統會是一個處理能力強大的大型伺服器，，然而實際上客戶端送來的服務要求係由許多伺服器組成的伺服器群組共同分擔服務，如此架構可以較低成本提升系統之處理能力。

在虛擬伺服器實作上可分為三種類型，分別是 1)網路位址轉換(Network Address Transform，NAT)技術；2)IP 隧道(IP Tunneling)技術；3)直接路由(Direct Routing，DR)等三種實作類型。由於本篇論文運用第三種方法直接路由技術改良原始 4S 架構之擴展上的缺陷，因此在本文中僅針對第三種方法做介紹。

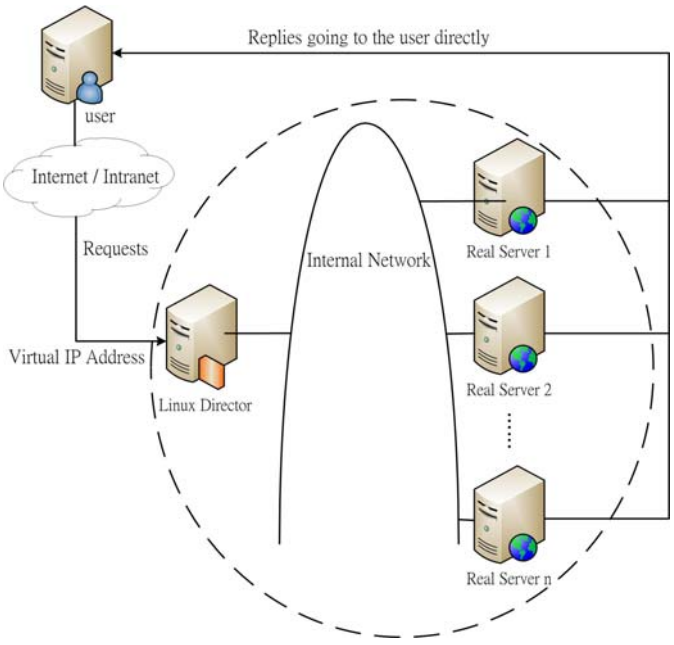

圖 4 Linux Virtual Server via Direct Routing

如圖 4所示，虛擬伺服器直接路由(LVS/DR)在實作上係使用 HUB 或 Switch 直接串接起來，虛擬IP位址則同時被負載平衡器(Linux Director)與所有的實體伺服器(Real Servers)共享。當負載平衡器收到客戶端的服務要求會依據預設的排程規則將要求透過實體線路送到對應的實體伺服器進行服務處理；當實體伺服器收到封包之後會當成本機內的封包進行處理，在完成服務要求後實體伺服器會直接透過對外的網路連線將回應送回客戶端。

虛擬伺服器在服務指派的演算法支援 1) Round-Robin (RR)、2) Weighted Round-Robin (WRR) 、 3) Least Connection (LC) 、 4) Locality-Based Least-Connection (LBLC) 、 5) Destination Hashing (DH)、6) Source Hashing (SH)、7) Shortest Expected Delay (SED)與 8) Never Queue (NQ)等八種排程演算法。

三、交替式安全伺服器架構介紹

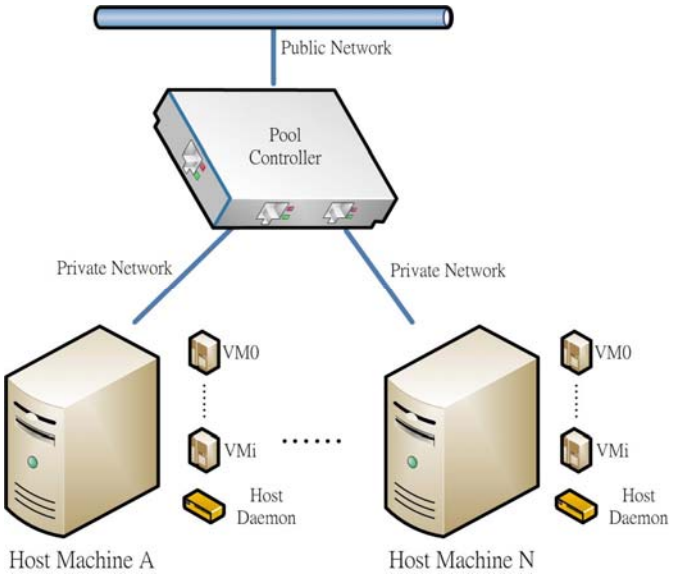

圖 5 原始 4S 系統架構圖

本研究所提出之交替式安全伺服器架構(Secure Server Switching System，簡稱 4S)透過冗置架構(Redundancy)達到容錯能力、並且周期性地將服務切換至不同的伺服器(Server)上限制入侵者的駐留時間，藉由跨主機的切換機制，在數台主機之間動態的平衡分配伺服器數量以有效的發揮所有處理能力。4S 系統架構圖如圖 5 所示，由伺服器群控制裝置(Pool Controller，PC)與數台實體電腦主機(Host)上建構之虛擬機器(Virtual Machine，VM)所構成的伺服器群(Server Pool)所組成：

(一) 伺服器群控制裝置(Pool Controller)

Pool Controller 是一個獨立的嵌入式系統裝置，其負責監控、蒐集所有實體主機(Host)與虛擬伺服器(Server)的運行狀態，並依據預訂好的演算規則與當時運行狀態動態的控制伺服器切換，對於客戶端傳來的封包僅依據排程演算規則作封包轉送之動作，並不會處理封包資料內容，假定 Pool Controller 不會遭受惡意攻擊者破壞。

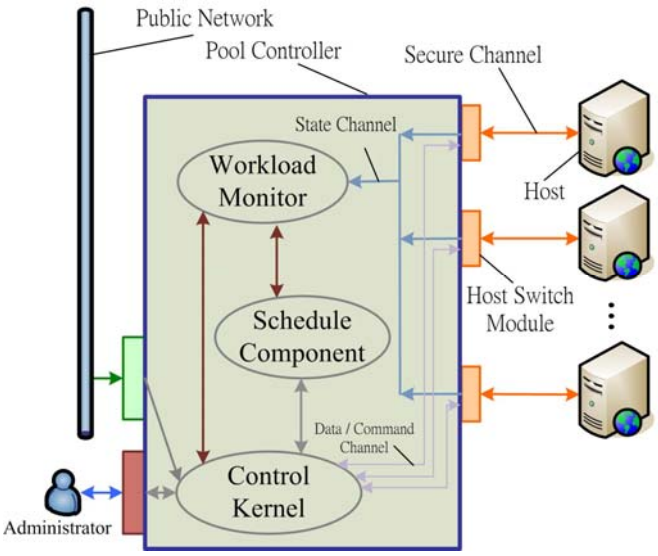

圖 6 伺服器群控制裝置(Pool Controller)架構圖

Pool Controller 內部架構圖參考如圖 6 所示，裝置內部由 Workload Monitor、Schedule Component 與 Control Kernel 三個主要單元組成：

Workload Monitor：

負責從各主機上的 daemon 蒐集主機與伺服器的即時運行狀態，即時的將運行資訊更新到 Pool Controller 的內部檔案結構。

Schedule Component：

依據 workload monitor 即時更新至 Pool Controller 內部檔案結構的資訊與預先定義好的排程演算規則以決定伺服器的上線與離線排程。

Control Kernel：

依據 schedule component 的排程資訊下命令給安裝於各台主機上的 daemon 管理伺服器。

(二) 主機(Host)

本研究採用 Virtual Box[10]作為虛擬機器監控系統(Virtual Machine Monitor，VMM)，以虛擬技術在數台主機上各自建立數台伺服器。並且在各台主機上安裝 Daemon 軟體，該軟體依循 Pool Controller 所傳遞之控制命令控制伺服器的上線與離線切換，並且周期性的自動傳遞本機與本機上運行的所有虛擬伺服器之即時狀態給 Pool Controller。

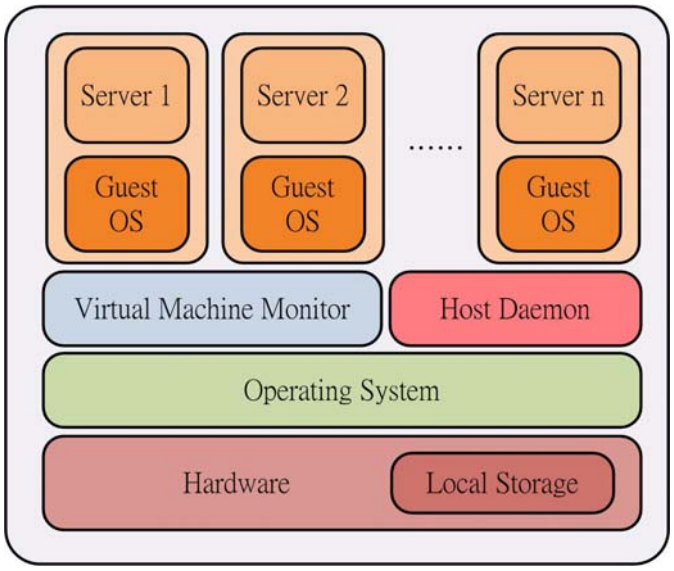

圖 7 主機端(Host)軟體堆疊圖

主機系統的堆疊關係如圖 7 所示，採用全虛擬化技術(Full-virtulization)在數台主機上建構出數台虛擬伺服器(Server 1, Server 2, …, Server N)，並且運用虛擬機器的快照功能(Snapshot)配合本系統架構的切換機制，讓進入離線的伺服器可以依據快照資料快速的將伺服器狀態置換回原本初始的狀態，如此可以確保即使在前一次服務周期遭到入侵所竄改的資料與後門設定在重新上線後均能將之完整的清除。

(三) 4S 架構的擴展設計

由於 4S 系統架構採用集中式的管理架構，並且對於服務工作的分配與回應亦集中在 Pool Controller 裝置上，雖然集中式的架構可以對整體伺服器群作整體性的監控與管理，但此設計將可能使 Pool Controller 成為系統未來擴展時的瓶

頸，因而限制了系統的擴充能力(Extensibility)。

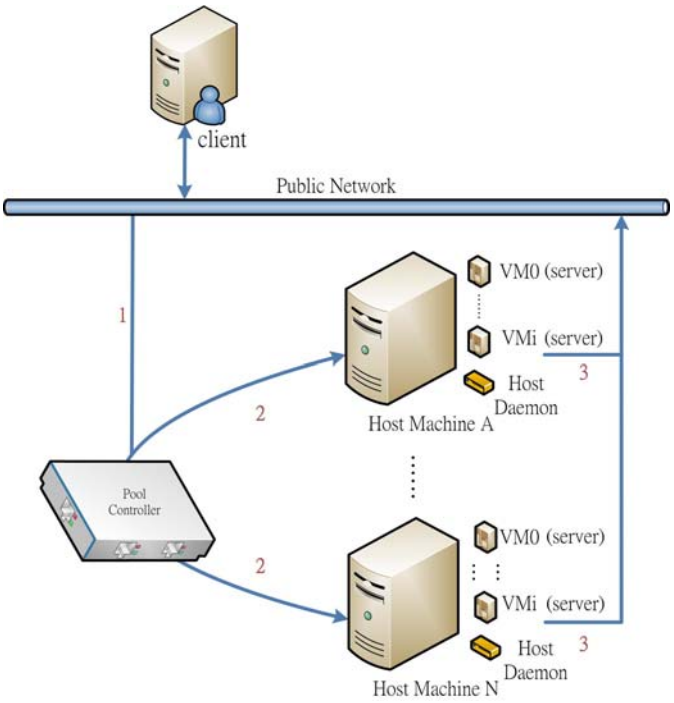

圖 8 改良後 4S 系統整體架構圖

為避免 Pool Controller 的網路頻寬成為系統架構擴展的阻礙，本研究提出一改進架構參考如圖 8 所示，其運作方式為：

1. Client 發出服務要求封包至系統上。
2. 由 Pool Controller 負責接收客戶送來的要求，依據排程結果以 direct routing 方式轉送給選定的 Server 進行服務處理。
3. 待 Server 完成服務處理後，將回應封包直接送回給 client。

運用 Linux Virtual Server 技術中的 Direct Routing (LVS / DR)技術將網路配置做些許調整，讓 Pool Controller 僅需負責監控資訊、排程管理與要求封包的轉送工作，原本伺服器處理完欲傳送回客戶端的資訊不再交由 Pool Controller 作轉送動作，而是直接透過各台主機對外的網路連線直接回傳給客戶端，如此將可大幅減少 Pool Controller 的工作量。

四、 系統實作與效能統計

4S 系統架構已完成設計，其實作環境介紹如下：伺服器群控制裝置以虹晶科技出產的 Cheetah Development Kit(CDK)系統晶片開發實驗板為平台；主機部分則由兩台 intel 雙核心主機搭配 4GB 記憶體，其主端作業系統(Host OS)安裝 ubuntu 8.04 LTS Desktop 版，VMM 採用 Sun xVM Virtual Box 3.0.2，分別在各台主機上設置四套客戶端作業系統(Guest OS)，其中客戶端作業系統安裝 ubuntu 9.04 Server 版本，虛擬伺服器環境設置為 384Mbytes 記憶體與 8.0Gbyte 之磁碟空間，並運行 Apache2 與 PHP5 網頁服務程序。另外，因本架構運用 direct routing 之協定，因此每台虛擬伺服器均須關閉其 ARP 之回應功能以解決 Server 與 Pool Controller 同時搶對外連線之 IP 位址。

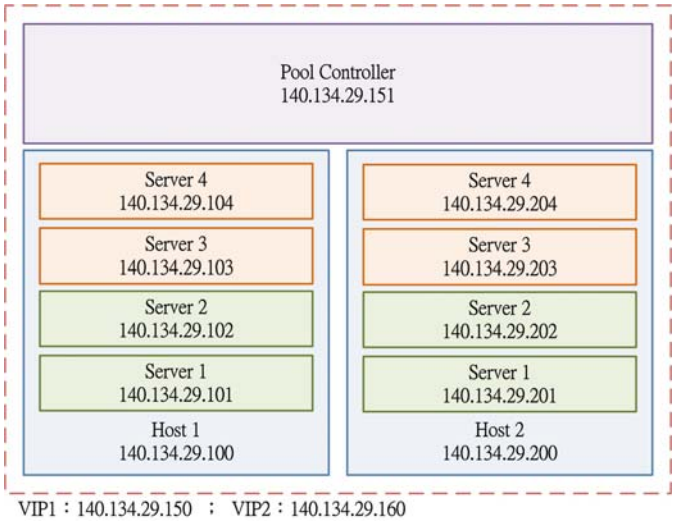

圖 94S 實作環境設定說明

參考圖 9，8 台虛擬伺服器分為兩組提供不同的相簿內容(VIP1 由 Host1 上的 Server1 與 Server2 及 Host2 上的 Server1 與 Server2 輪替切換服務；VIP2 由 Host1 上的 Server3 與 Server4 及 Host2 上的 Server3 與 Server4 輪替切換服務)並且運行時伺服器將平均分散在兩台主機上。依據分組在 Server 上線後將監聽 VIP1 或 VIP2 位址之封包，但僅會對由 Pool Controller 轉送之封

包進行接收處理，最後結果將經由網域之路由器回送到客戶端。

在正常情況下，提供的網頁服務會由 8 台伺服器輪替上線服務，並且未上線服務的伺服器會依據快照內容還原伺服器的初始狀態。在面對伺服器可能遭受的攻擊與非預期之故障，我們做出以下的假設狀況測試：

伺服器資料遭到竄改：我們在切換周期內假裝為入侵者登入上線服務的伺服器，並將原本提供的風景照片置換為模特兒寫真照片以此假設入侵者成功竄改伺服器內容。在我們預設的置換周期 30 秒一到系統完成伺服器輪替切換後，遭竄改的資料內容又恢復原先的風景照片。

主機故障或部分網路損毀：我們隨機挑選拔掉一台主機的網路連接線，以此假設實體主機的硬體故障或部分網路損毀的情況，在 Pool Controller 偵測到主機停止服務事件後，隨即將原先由該主機上伺服器提供的服務交由其他正常工作的伺服器接手接續服務，以維持系統服務的可用性。

```
**********************************************
[Host1] IP: 140.134.29.100      state: Normal
        VM1 State:Ready VM2 State:Working
[Host2] IP: 140.134.29.200      state: Normal
        VM1 State:Ready VM2 State:Normal
[Host1] IP: 140.134.29.100      state: Normal
        VM3 State:Ready VM4 State:Ready
[Host2] IP: 140.134.29.200      state: Normal
        VM3 State:Ready VM4 State:Working
==============================================
[INFO]Switching to Host2/VM1
[INFO]Switching to Host1/VM3
[ACTION]Sending message: 0x01020003 to Host1
[ACTION]Sending message: 0x02040003 to Host2
**********************************************
[Host1] IP: 140.134.29.100      state: Normal
        VM1 State:Ready VM2 State:Offline
[Host2] IP: 140.134.29.200      state: Normal
        VM1 State:Working       VM2 State:Ready
[Host1] IP: 140.134.29.100      state: Normal
        VM3 State:Working       VM4 State:Ready
[Host2] IP: 140.134.29.200      state: Normal
        VM3 State:Ready VM4 State:Offline
```

圖 10 伺服器群控制裝置即時監測訊息(1)

故障主機或損毀網路修復：接續上一假設，我們將網路連接線重新接上假設故障主機或損毀網路已完成修復並將重新上線服務，待 Pool Controller 偵測到新的／修復完成的主機後，會自動將(重)新加入的伺服器納入服務候選名單，並依據預先定義好的分散伺服器至不同主機的原則將伺服器平均分配在各台主機上，確保工作負載的平衡性。

如圖 10 所示為 Pool Controller 運行時的監控畫面內容，畫面顯示在固定時間內偵測各個伺服器與主機的運行狀況並顯示在螢幕上，在置換周期時間屆滿時進行服務伺服器的置換：首先，VIP1 相簿服務由 Host1 上的 Server2(即 VM2)提供服務，而 VIP2 相簿服務由 Host2 上的 Server4(即 VM4)提供服務；待切換周期屆滿，Pool Controller 確定 Host 與 Server 狀態正常，依據預定義好的排程規則挑選下一組服務伺服器 Host2 上的 Server1 與 Host1 上的 Server3 提供接續的服務，接著發送訊息給 Host1 與 Host2 上的 deamon 要求下線後的 Server 進行快照回復程序。

```
**********************************************
[Host1] IP: 140.134.29.100      state: Normal
        VM1 State:Ready VM2 State:Ready
[Host2] IP: 140.134.29.200      state: Normal
        VM1 State:Ready VM2 State:Working
[Host1] IP: 140.134.29.100      state: Normal
        VM3 State:Ready VM4 State:Working
[Host2] IP: 140.134.29.200      state: Normal
        VM3 State:Ready VM4 State:Ready
**********************************************
[Host1] IP: 140.134.29.100      state: Normal
        VM1 State:Fail  VM2 State:Fail
[Host2] IP: 140.134.29.200      state: Normal
        VM1 State:Ready VM2 State:Working
[Host1] IP: 140.134.29.100      state: Normal
        VM3 State:Fail  VM4 State:Fail
[Host2] IP: 140.134.29.200      state: Normal
        VM3 State:Ready VM4 State:Ready
[ERROR]Host1 suspended...wait one more cycle
[INFO]Switching to Host2/VM3
**********************************************
[Host1] IP: 140.134.29.100      state: Normal
        VM1 State:Fail  VM2 State:Fail
[Host2] IP: 140.134.29.200      state: Normal
        VM1 State:Ready VM2 State:Working
[Host1] IP: 140.134.29.100      state: Normal
        VM3 State:Fail  VM4 State:Fail
[Host2] IP: 140.134.29.200      state: Normal
        VM3 State:Working       VM4 State:Ready
```

圖 11 伺服器群控制裝置即時監測訊息(2)

圖 11 為模擬主機故障過程的 Pool Controller 運行監控擷取之畫面，一開始 VIP1、VIP2 相簿分別由 Host2 的 Server2 與 Host1 的 Server4 提供服務；接著 Pool Controller 偵測到 Host1 的所有 Server 均無回應隨即註記 Host1 為 suspend 狀態表示該主機暫不列入排程候選名單，並且發現 VIP2 相簿的服務提供伺服器已無回應，因此立即將服務切換至 Host2 上的 Server3 繼續提供服務。

在網頁服務效能的測量上，我們建構一台獨立主機並採用 apache benchmark(ab)在固定時間內對網頁送出大量服務要求，並檢視系統對於服務的回應時間(Response time)作為效能優劣的評估依據。

測試環境之參數設定在一秒內會有 500 個使用者發出瀏覽要求，並且每位使用者同時發出的服務要求數量為 10 筆。

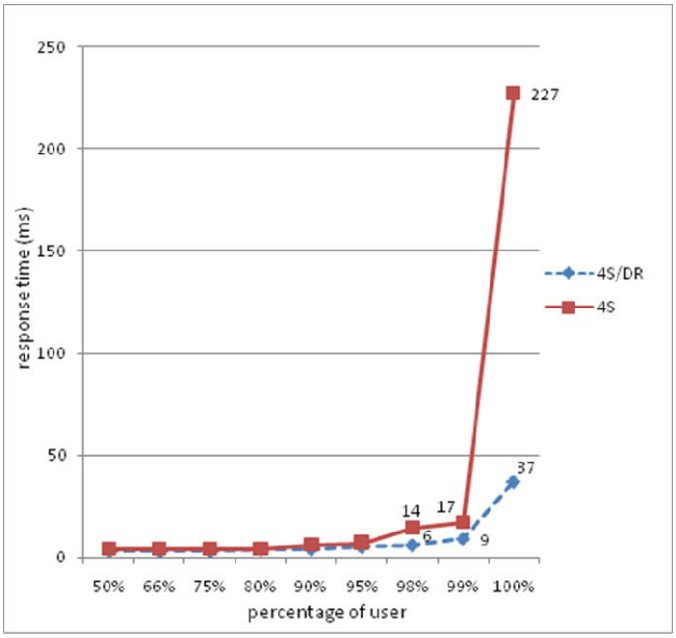

圖 12 效能比較

圖 12 表示在本測驗中 500 個使用者感受到伺服器回應速度的折線圖，橫軸為連線人數的比例，縱軸為使用者感受到的反應時間(單位為毫秒，ms)。紅色實線條為初始 4S 架構之量測數據，在初始架構中可以確保 99%大部分的使用者要求能在 17 毫秒內完成，但有約 1%的使用者其回應時間將拉長至 227 秒；藍色虛線條為採用 direct routing 協定之架構，此架構可以確保 99%的使用者服務能在 9 毫秒內完成，甚至所有使用者的服務要求都能在 37 毫秒內完成。此一測試充分證明 4S 架構在採用 LVS/DR 的路由技術後，已能免除多重伺服器通路瓶頸的阻塞狀況，使 4S 系統能透過 pool controller 同時控制許多主機／伺服器，充分發揮系統擴充及負載平衡的效益。

五、結論

本研究在實作上已完成伺服器的切換控制，並且可以動態更改伺服器之排程數量，亦即在 4S 開始週期性的交替運行過程中可以動態的增減伺服器數量，甚至是將一整台主機退出服務亦能維持服務的正常運行，並確保重新將主機上線後亦可馬上加入排程提供服務。

4S 架構確保主機在動態增加／移除的過程中服務能正確無誤的運行具備高度延展性，並且對於硬體的故障亦能透過 Pool Controller 調整切換將服務即時導向正常的伺服器繼續運行，具備高可用性。

最後為避免 Pool Controller 在日後成為系統擴展之絆腳石，運用 Direct Routing 之協定並更改伺服器與對外網路及 Pool Controller 之連線關係以減輕伺服器群控制器之工作負擔，由實驗量測得到之數據可以證明採用 direct routing 確實能有效地降低中央控制裝置的負載。

除完成上述討論項目外，本研究將繼續針對工作負載平衡演算機制進行改良，讓伺服器切換演算機制能夠確實考量各台主機之運算能力與即時的資源使用狀況…等因素，讓平衡不單只是伺服器數量上的平均分配，而是考量運算處理能力的平均分配。

六、參考文獻

[1] Amit Singh, “An Introduction To Virtualization, “ http://research.ihost.com/osihpa/osihpa-hensbergen.pdf, February 5, 2004.

[2] Anantha K. Bangalore and Arun K Sood, “Securing Web Servers Using Self Cleansing Intrusion Tolerance (SCIT)”, DEPEND, June 2009.

[3] David Arsenault, Arun Sood, and Yih Huang, “Resilient Computing Clusters: Self-Cleansing Intrusion Tolerance with Hardware Enforced Security (SCIT/HES),” Proceedings Second International Conference on Availability, Reliability and Security (ARES 2007), Vienna, Austria. pp. 343-350, April 2007.

[4] J. E. Smith and R. Nair, “The Architecture of Virtual Machines,” IEEE Computer, 38(5):32–38, May 2005.

[5] M. Rosenblum, and T. Garfinkel,“Virtual Machine Monitors: Current Technology and Future Trends,” Vol. 38, No. 5, IEEE Computer, pp. 39-47, 2005 May.

[6] Nikhil Bhatia, Jeffrey S. Vetter, “Virtual Cluster Management with Xen.” Lecture Notes in Computer Science, Volume 4854/2008, pp. 185-194, 2008.

[7] Quyen Nguyen and Arun Sood, “Quantitative Approach to Tuning of a Time-Based Intrusion-Tolerant System Architecture”, 3rd Workshop on Recent Advances in Intrusion Tolerant Systems, Portugal, June 29, 2009.

[8] R. P. Goldberg. “Architecture of virtual machines,” In proc. of the workshop on virtual computer system s, NY, USA, pages 74-112. ACM Press , New York, 1973.

[9] Symantec, “Symantec Internet Security Threat Report,” Volume XII, September 2007.

[10] VirtualBox, Sun Microsystems, Inc., http://www.virtualbox.org/

[11] Wang Yu-Sheng, Wang Lei, “Secure Server Switching System,” Cryptology and Information Security Conference (CISC), March 2009.

[12] Wensong Zhang, Shiyao Jin, and Quanyuan Wu, Creating Linux Virtual Servers, The 5th Annual Linux Expo Conference, May 1999.

[13] Yanping Gao, Xiangjun Li and Yandong Che, “New Architecture and Algorithm for Webserver Cluster based on Linux Virtual Server,” International Symposiums on Information Processing, pp. 520-524, May 2008.

[14] Yih Huang, Arun Sood, and Ravi K. Bhaskar, “Countering Web Defacing Attacks with System Self-Cleansing,” Proceedings of 7th Word Multiconference on Systemics, Cybernetics and Informatics, Orlando, Florida, pp. 12-16, 2003 July.

[15] Yih Huang, David Arsenault, and Arun Sood, “Incorruptible Self-Cleansing Intrusion Tolerance and Its Application to DNS Security," Vol. 1,No. 5, Journal of Networks, pp. 21-30, 2006 Sep-Oct.