雲端虛擬桌面系統之分散式架構及負載平衡改進

湯凱崴
財團法人資訊工業策進會
雲端系統軟體研究所 雲端平台研發中心
kwtang@iii.org.tw

摘要

資策會的雲端個人電腦（Personal Cloud Computer，PC2）是一跨平台的雲端虛擬桌面系統，此系統的核心基於遠端桌面（Remote Desktop）技術。一般遠端桌面是建置在固定的實體或虛擬伺服器之上，若直接將此架構導入大型雲端桌面系統，易使系統在多使用者同時連入時發生伺服器過載的問題。因此我們提出一個基於分散式架構的改進方法，目標是確保雲端桌面系統的負載平衡，避免負載量集中在少數伺服器之上。

關鍵詞：雲端運算、遠端桌面、負載平衡、分散式系統

1. 前言

由於系統架構不同，程式難以執行於異質平台。一般為 Windows 開發的應用程式無法在 Linux 平台中執行，而 Linux 程式亦無法被執行於 Windows 平台。此問題在行動裝置上依然存在，使用者無法以平板電腦或智慧型手機執行傳統桌面程式。若企業欲提供完整的行動辦公體驗，讓員工使用行動裝置仍能執行其熟悉的桌面程式，則須投入許多成本來進行軟體移植。此外必須維護多套程式碼，方能同時顧及 Android、iOS 等各大主流行動作業系統。欲解決平台的問題，採用遠端桌面技術是較低成本的解決方案。藉由常見的遠端桌面軟體遙控遠端電腦，以操作遠端電腦上的應用程式。使用此方法，軟體無需重新改寫即可供不同平台的用戶使用。而行動裝置的使用者亦可以安裝遠端桌面 App，實現行動辦公之目的。

要提供遠端桌面服務並不難，大部分作業系統已內建相關功能，一般個人或組織皆能輕易架設遠端桌面伺服器。然而，以此方式架設的遠端桌面是由多個使用者共用一台遠端電腦，當大量使用者同時連入此伺服器，易使該電腦過載而發生不可預期的結果。為解決此問題，我們提出一個改進方法，將遠端桌面動態分散至各伺服器執行，以提升大型遠端桌面系統的穩定性。此方法目前已實際使用於資策會的雲端系統中。

本文後續章節內容編排如下：在第二節中，我們將介紹幾個常見的遠端桌面技術並回顧相關研究；第三節將探討傳統遠端桌面架構所存在的問題，並提出改進方法；而第四節將說明此方法於資策會雲端系統的實際應用，以證明其可行性及價值；第五節將總結此研究；第六節為本研究的未來研究方向。

2. 遠端桌面技術及相關研究

RDP 及 VNC 是常見的兩類遠多桌面技術，相關名詞介紹如下：

- **RDP**：RDP[1]的全名為 Remote Desktop Protocol，是微軟公司為遠端桌面連線開發的通訊協定，大部分 Windows 作業系統皆已內建 RDP 相關模組。RDP 採用 Client-Server 架構，預設使用 3389 連接埠。此協定能將遠端電腦（server）的螢幕畫面及聲音即時傳送到客戶端（client），並將客戶端的滑鼠、鍵盤等使用者動作送回遠端伺服器，以達到遙控遠端電腦的目標。透過 RDP，用戶打開客戶端軟體，輸入伺服器位址、帳號、密碼等資訊即可連入遠端電腦，以使用遠端的程式及資料。目前 RDP 已發展到 8.0 版本，並內建於 Windows 8 及 Windows Server 2012 等作業系統[2]。
- **FreeRDP、Rdesktop 及 XRDP**：RDP 官方版本僅支援 Windows 及 Mac[3]系統，在 Linux 上需由第三方套件實現。FreeRDP[4]及 Rdesktop[5]是開放原始碼的 RDP 客戶端軟體，可用以連入遠端的 RDP 伺服器；而伺服器的部份則可由 XRDP[6]來實現，在架設完 XRDP 後，即可以 RDP 客戶端遙控 Linux 的 X Window 桌面。
- Seamless RDP：Seamless RDP[7]是一基於 RDP 的遠端桌面技術。不同於 RDP 投放整個桌面畫面到客戶端，Seamless RDP 僅傳輸遠端桌面中的執行中的程式之畫面到用戶的裝置。由於不需傳送整個遠端桌面，Seamless RDP 所佔用的網路頻寬低於傳統 RDP。
- **VNC**：為 Virtual Network Computing[8]的縮寫。相對於微軟開發的 RDP，VNC 是另一類常見的遠端桌面技術，該技術基於 RFB (Remote Framebuffer)[9]協定。類似於 RDP，RFB 也是一個 client-server 架構的遠端桌面協定。一般常見的系統如 Windows、Linux、Mac 等，RFB 皆有支援。VNC 軟體採用 GPL 授權條款，目前已發展出許多著名的分支，如 TigerVNC[10]、UltraVNC[11]、RealVNC[12]、Vinagre[13]…等等。

雲端運算是遠端桌面重要的應用，由於執行於雲端的虛擬系統無實體螢幕，使用者常以遠端桌面軟體遙控雲端系統。Manvar 等人[14]的研究表示

桌面系統的雲端化可提高資源使用率並使系統更易管理；Kochut 等人[15]曾進行桌面系統的效能分析，旨在改善桌面雲端系統的效能。而在行動運算的領域，行動裝置對桌面電腦的遠端連線亦有不少研究成果，如 Liao 等人[16]的研究指出遠端桌面技術可解決行動裝置運算效能不足的問題；Lin 等人[17]曾研究 VNC 及 RDP 等技術對行動裝置耗電量的影響；Khankan 等人[18]進一步探討了行動裝置連入遠端電腦的各種方法，並分析這些方法的優劣……等等。

3. 遠端桌面之改進－導入分散式架構及負載平衡機制

在傳統的遠端桌面系統，使用者帳號與伺服器之間是相互綁定的，亦即伺服器提供一個或多個可供遠端連入的使用者帳號，使用者只能連入該帳號所屬的伺服器。在此架構下，若同時連入某伺服器的使用者（Concurrent User）數目超過伺服器所能承受的上限，將導致此伺服器負載量過大而無法正常提供服務。此外，當伺服器損壞或者停機維修時，該伺服器下所有使用者皆無法登入系統。

改採分散式架構並使用負載平衡（Loading Balance）機制可解決上述問題。亦即，使用多台伺服器構建遠端桌面系統，並將運算工作平均分配到各個伺服器，避免單一伺服器的負載量過高。為達到此目的，首先必須使桌面能被系統動態建立及移除，讓使用者的遠端桌面可在各伺服器間任意移動。然而在現有架構下，管理者須事先於伺服器創建帳號，使用者才能遠端登入該伺服器。欲在任意伺服器上執行同一使用者的遠端桌面，在此架構下較難實現。

我們在各個提供遠端桌面連線的伺服器上皆佈署一支常駐背景服務，負責動態產生、移除使用者的桌面環境。而使用者的檔案則以 Samba 儲存在獨立的檔案伺服器（File Server），避免資料隨著桌面環境的移除而消失。在使用者登入後，系統從檔案伺服器讀出使用者資料，還原使用者上次登出前的工作環境，並動態產生該使用者的遠端桌面環境。

接著導入分散式架構，遠端桌面不再執行於單一伺服器，而是分散在各個伺服器執行。在使用者登入後，系統於多台伺服器上準備該使用者的桌面環境，這些伺服器分工執行使用者所啟動的各個遠端程式，最後藉由 Seamless RDP 技術，將所有伺服器上該用戶執行中的程式畫面傳送到其終端裝置，於客戶端組合成一個完整的虛擬遠端桌面。

圖 1 是一個運行中的 Windows 8 遠端桌面，我們以此例來解釋上述架構。圖中使用者在遠端桌面上開啟兩個程式，分別為 Visual Studio 及 Internet Explorer。改採分散式架構後，這兩個程式可分散至兩台伺服器執行，一台執行 Visual Studio，另一台執行 Internet Explorer。並透過檔案伺服器確保使用者資料在各伺服器間的一致性。這兩個程式的螢幕畫面會在終端裝置上組合起來，故能維持與舊架構相同的遠端桌面使用體驗。藉此桌面虛擬化機制，使用者的遠端桌面可執行於多個伺服器之上，不再是綁定在固定伺服器中。

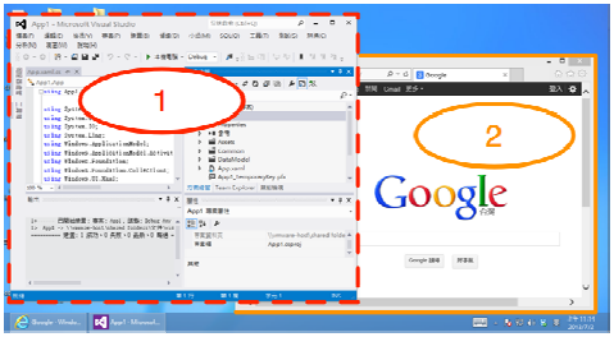

圖 1：使用者可在遠端桌面中開啟多個應用程式。

調整為分散式架構後，可進一步導入負載平衡機制。CPU 使用率、剩餘記憶體、目前連線數目（sessions）為是本研究考量的伺服器負載因子。對於系統中任一伺服器 x，我們使用$Priority(x)$來決定系統挑選該伺服器的優先順序：

$$\begin{aligned} Priority(x) = \{&[100 - cpu_c(x)] \cdot W_{CPU}\} \\ &+\{[ram_m(x) - ram_c(x)] \cdot W_{RAM}\} \\ &+\{[sessions_m(x) - sessions_c(x)] \\ &\cdot W_{sessions}\}, \\ &\forall x: x \in servers \end{aligned} \quad (1)$$

公式中的符號定義如下：

- $cpu_c(x)$、$ram_c(x)$、$sessions_c(x)$：分別代表伺服器 x 目前的 CPU 使用率、記憶體使用量以及目前連入的使用者數目；
- $ram_m(x)$、$sessions_m(x)$：分別代表伺服器 x 最大記憶體使用量以及最大允許的連線數目。由於 CPU 使用率最大為 100，故不另行定義 $cpu_m(x)$。在最大連線數方面，我們設定伺服器上每 1G 的記憶體，可接受最多 20 個連線，亦即：

$$sessions_m(x) = (ram_m(x) / 1024) \cdot 20 \quad (2)$$

- W_{cpu}、W_{ram}、$W_{sessions}$：為常數值，分別用以控制 CPU 使用率、記憶體、連線數三項因子的權重。權重值越大，該因子對負載平衡演算法的影響力越高。在一般情況下，該值設定為 1 即可。

依據公式 1，我們制定圖 2 的演算法。圖中 *app* 為一個使用者欲執行的遠端程式，*serversList* 為系統中可用伺服器之清單。首先需計算各伺服器的 Priority(x)值，並根據其值由大到小排序伺服器清單。接著依序掃描這些伺服器上的應用程式，若伺服器 *serv* 裝有程式 *app*，代表該伺服器可用以執行 *app* 程式，故回傳 *serv*。若直到迴圈結束仍未找到合適的伺服器，表示 *app* 程式不存在於系統中，即傳入的參數有誤，故回傳 *ERROR*。在執行完這個演算法後，系統將得到一台裝有 *app* 且

Priority(*x*)值最大的伺服器，接著即可在該伺服器上執行 *app*。根據公式 1，伺服器負載量越低，其 Priority(x)值越大，故此演算法可保證每個應用程式皆被執行於當時負載量較低的伺服器中，進而達到負載平衡之功效。

```
chooseServer(serversList, app)
{
      buildPriority ( serversList )

      // sort descendingly
      sortByPriority ( serversList )

      for (i=0 to serversList.Length-1 )
      {
            serv = serversList[i]

            // find a server to run app
            if (app is installed on serv)
            {
                  return serv
            }
      }
      return "ERROR"
}
```

圖 2：負載平衡機制－以演算法挑選合適的伺服器。

圖 3 是本方法的架構圖。在一個採用本分散式架構的遠多桌面系統中，當使用者點擊一個遠端桌面上的應用程式圖示，系統將從可用伺服器中選擇一台負載量較低且裝有該程式的伺服器，在此伺服器上執行使用者所點擊的程式，並將程式畫面傳回給使用者。而檔案伺服器則用以儲存使用者的所有檔案。由於桌面上的程式並不執行於同一伺服器，當使用者開啟大量程式，或者多使用者同時登入時，此機制可降低單一伺服器過載的機率。

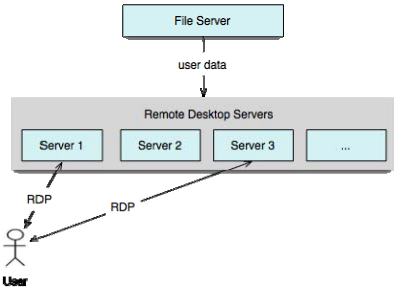

圖 3：將遠端桌面系統拆解為分散式架構。

4. 於資策會雲端系統之實際應用

PC2（Personal Cloud Computer）是財團法人資訊工業策進會研發的雲端軟體系統，其定位為中小企業私有雲。透過該系統，應用程式不再受限於原生平台，使用者可使用任意作業系統，透過瀏覽器隨時隨地使用雲端上的程式。在 PC2 系統上，任何程式皆執行於雲端伺服器，再以 RDP 技術即時傳送畫面至使用者的裝置。PC2 能將這些遠端程式模擬為本機程式，在網路順暢的環境，可達到近似於本機裝置執行的效果。

圖 4 是 PC2 的運行畫面，每個使用者登入後皆有一個專屬的虛擬桌面，這部份並不使用遠端桌面技術傳送，而是採 HTML 呈現在使用者的瀏覽器以節省網路頻寬。在虛擬桌面上，管理者授權的程式將呈現於此。使用者可點擊桌面上的應用程式圖示，直接在雲端上執行這些程式。系統將程式視窗以 RDP 技術即時顯示至使用者的螢幕，並運用 Java Applet 讓這些程式出現在使用者電腦的工作列上，以達到模擬成本機程式的效果。依據系統的調配，這些程式可能被執行於不同伺服器上，再個別與使用者的電腦進行 RDP 連線。而除了傳統電腦，PC2 亦可執行於 iOS、Android 等行動裝置上（圖 5）。使用 PC2 的客戶端 App，企業或學校的程式軟體無需移植即可被執行在行動裝置之上，節省可觀的技術成本。

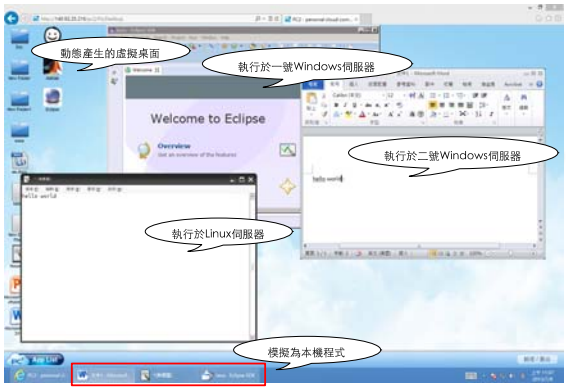

圖 4：PC2 客戶端的執行畫面。

PC2 系統由中央伺服器（Session Manager）、應用程式伺服器（Application Server）及檔案伺服器三個部分組成。其架構如圖 6 所示，分別說明如下：

- **中央伺服器**：負責驗證使用者帳號、管理用戶權限、維持系統的負載平衡、協調各伺服器等工作；
- **應用程式伺服器**：即上一節提到的遠端桌面伺服器，用以執行被使用者啟動的遠端程式，再將畫面以 RDP 技術投放到使用者的裝置。PC2 系統包含一台或多台應用程式伺服器，這些伺服器可為 Windows 電腦或 Linux 電腦，若使用者所開啟之程式為 Windows 程式，則在 Windows 電腦執行，反之則在 Linux 上執行；
- **檔案伺服器**：保存使用者的檔案以及使用者桌面的環境設定。

當使用者要求登入 PC2 系統，首先連入的是中央伺服器，此伺服器會進行使用者帳號的合法性驗證。若帳號及密碼正確，則從檔案伺服器中讀出使用者的桌面環境設定，並從應用程式伺服器中讀出可用程式之清單，藉由這些資訊動態產生一個虛擬桌面。若採用傳統 RDP 機制，在此虛擬桌面上被啟動的程式會全部被執行在單一伺服器中。

當該使用者在桌面執行大量遠端程式，或者多個用戶同時登入此伺服器的 PC2 服務時，該伺服器易因負載量過高而不穩定。故我們使用本文的方法，讓系統自動安排程式在合適的應用程式伺服器中執行，以確保各個伺服器皆能穩定地提供 PC2 服務。

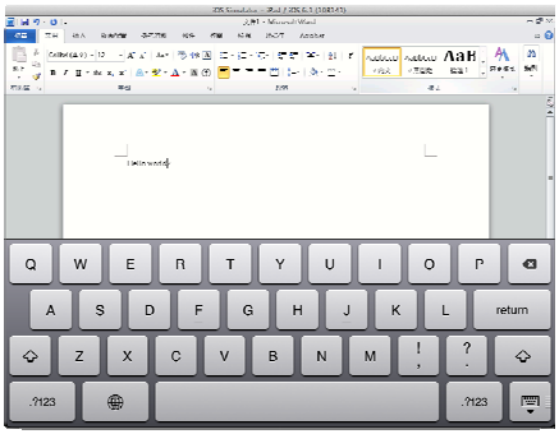

圖 5：透過 PC2，行動裝置亦可執行傳統桌面軟體。

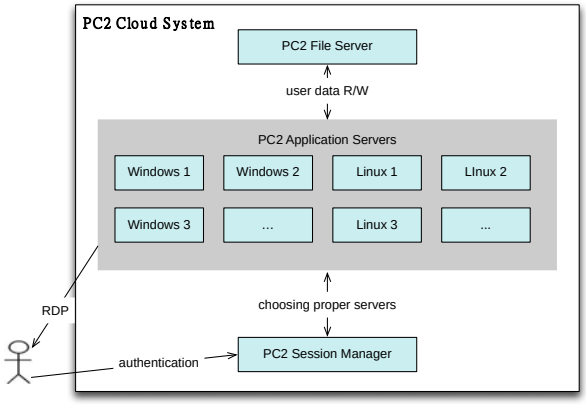

圖 6：PC2 系統由中央伺服器、應用程式伺服器及檔案伺服器組成。

表 1 是一個實際營運中的 PC2 系統之伺服器資訊。此系統運行於資策會雲端研究所，主要供該部門內部員工使用，其使用者數約 90 人。該系統由四台應用程式伺服器組成，我們分別以 A、B、C、D 稱之。其中，伺服器 A 為 Ubuntu 10.04 作業系統，負責執行 Linux 程式，並肩負中央伺服器及檔案伺服器的工作；伺服器 B、C、D 為 Windows Server 2008 R2 作業系統，負責執行 Windows 程式。圖 7 是此 PC2 系統在 2013 年 7 月 15 日至 2013 年 7 月 31 日期間各伺服器的連線數所佔百分比（分子為伺服器連線數，分母為系統總連線數），觀察此圖可發現使用者的連線大致是平均分布在各伺服器之上，而非集中在單一機器。

表 1：實驗伺服器之基本資訊。

伺服器	角色	作業系統
A	PC2 應用程式伺服器 PC2 中央伺服器 PC2 檔案伺服器	Ubuntu 10.04
B	PC2 應用程式伺服器	Windows Server 2008 R2
C	PC2 應用程式伺服器	Windows Server 2008 R2
D	PC2 應用程式伺服器	Windows Server 2008 R2

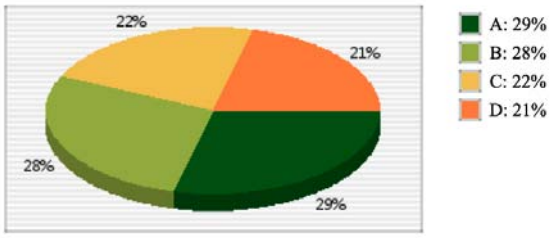

圖 7：各伺服器連線數所佔百分比。

5. 結論

遠端桌面應用廣泛，藉由該技術，使用者可隨處存取遠端電腦的程式及資料。此外，遠端桌面亦可解決軟體難以跨平台的問題。傳統的遠端桌面服務由單一伺服器提供，在大型雲端系統上，此架構的負載能力面臨挑戰。本研究提出一個改進架構，使遠端桌面上的程式得以分散在各個伺服器執行。並導入負載平衡機制，減低單一伺服器過載的機率。總結本文，此研究分為三部分：

1. 遠端桌面環境的動態佈署；
2. 將遠端桌面切割為數個運算區塊，分散到多台伺服器執行；
3. 評估 CPU 使用率、剩餘記憶體、連線數等因子，並分析各伺服器可執行的程式，使上述運算區塊能被平均執行在各個伺服器，確保系統整體的負載平衡。

實務上，此研究實作於資策會的 PC2 雲端桌面系統。當大量使用者同時湧入 PC2 時，本架構能協助系統平均使用各伺服器的硬體資源，以達到負載平衡之目標。

6. 未來研究方向

未來我們將嘗試預測各種程式的資源使用率，由此資訊動態調整本文公式中的權重值參數。例如，若預測某程式將消耗較高的運算量，則在演算法中提高 CPU 權重，使系統加強考慮具較多閒置運算量的伺服器。此外，我們將評估更多影響

系統負載能力的因素，例如伺服器的硬碟 I/O 頻率，以期發展出更好的雲端桌面負載平衡機制。

參考文獻

[1] “White Paper: RDP Features and Performance”, Microsoft, 2010.

[2] “Remote Desktop Protocol”, Wikipedia (http://en.wikipedia.org/wiki/Remote_Desktop_Protocol).

[3] “Microsoft Remote Desktop Connection Client for Mac”, http://www.microsoft.com/mac/remote-desktop-client.

[4] “FreeRDP”, http://www.freerdp.com.

[5] “Rdesktop”, http://www.rdesktop.org.

[6] “XRdp”, http://www.xrdp.org.

[7] Seamless RDP, http://www.cendio.com/seamlessrdp.

[8] T. Richardson, Q. Stafford-Fraser, K.R. Wood; A. Hopper, “Virtual Network Computing”, IEEE Internet Computing, Vol. 2, No. 1, 1998

[9] T Richardson, “The RFB Protocol specification”, RealVNC Ltd, November, 2010.

[10] “TigerVNC”, http://sourceforge.net/apps/mediawiki/tigervnc.

[11] “UltraVNC”, http://www.uvnc.com.

[12] “RealVNC”, http://www.realvnc.com.

[13] “Vinagre”, http://projects.gnome.org/vinagre.

[14] D Manvar , M. Mishra, A. Sahoo, “Low Cost Computing Using Virtualization For Remote Desktop”, Fourth International Conference on Communication Systems and Networks, 2012, Bangalore, Indian.

[15] A. Kochut, K. Beaty, H. Shaikh, D.G. Shea, “Desktop Workload Study with Implications for Desktop Cloud Resource Optimization”, IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum, 2010, Atlanta, USA.

[16] X. Liao, H. Jin, Z. Bao, H. Li, “MoClouDesk: Virtualized Desktop for Mobile Environment”, IEEE 14th International Conference on Computational Science and Engineering, 2011, Dalian, China.

[17] Y. Lin and M.D. Francesco, "Energy Consumption of Remote Desktop Access on Mobile Devices: An Experimental Study", IEEE 1st International Conference on Cloud Networking, 2012, Paris, France.

[18] K. Khankan and R. Steele, “Challenges for Mobile Remote Access to Desktop and Web Resources”, Proceedings of the 3rd International Conference on Mobile Technology, Applications & Systems, 2006, New York, USA.

7. 致謝

本研究依經濟部補助財團法人資訊工業策進會「 102 年度雲端運算系統及軟體技術研發計畫 (2/3) 」辦理。