Web Application 跨網域權限存取控制之設計

蔡政宇
南華大學
資訊管理所
sfw.sakana@gmail.com

林育弘
南華大學
資訊管理所
u0111095@yahoo.com.tw

楊惠媚
大同技術學院
企業管理系
nancy@ms2.ttc.edu.tw

王昌斌
南華大學
資訊管理所
cbwang@mail.nhu.edu.tw

摘要

隨著資訊應用複雜性，及開發成本考量，虛擬團隊合作開發，已成為趨勢之一。現今許多網頁應用程式(Web Application)的設計，都是由虛擬團隊分工設計而成；而這些開發的網頁應用程式，都會建構會員管理機制，用以區分閱覽者(Guest)、一般會員(Member)、管理者(Admin)。在權限控管的部分，較多採用以角色為基礎的存取控制(RBAC, Role-Based Access Control)，作為存取控制的基礎。

這個機制在實作上受到居多的限制，如 Session 無法跨越到另外一個 Application 使之存取；採 Cookies 的方式會受到必須為相同網域的限制；大部分的解決方案是將所有的 Web Application 統合在同一台伺服器上，以方便實作整個系統對於所設計的 Web Application 施行存取控制。但這個方法會給虛擬團隊的開發者帶來困擾，團隊群必須把 Web Application 建構在統一的伺服器上，開發者除了要在本地端撰寫完 Application 後，再上傳至伺服器，在這之間有著許多同步共構等問題。

為解決上述問題，本研究以 Session 結合 XML 達成，Web Application 跨域網域/伺服器的權限存取控制實作，使各個網頁應用程式可以在自己的伺服器中運作，而不需統一彙整到同一伺服器中運行，且不再受限整個系統中網頁應用程式需以同一程式語言撰寫。

關鍵詞：RBAC、Access Control、XML、跨網域

Abstract

With the complexity of IT applications, and costs considerations, virtual teamwork development, has become the trend. Many of today's web applications design, by the virtual design team from the division, and the development of these web applications, will build membership management mechanism to guest, the general membership (Member), Manager (Admin). Competence in the control part of a more Role-Based Access Control, as the basis for access control.

It is for this mechanism on the subject to many restrictions, Application Session only for a role and cannot leap to another Application Access; If the way to stop cookies for access, although solve the above problem, but cookies can only be accessed with a domain restrictions; The most simple solution is the yield under this restriction, all the Web Application integration on the same server to facilitate for the system as a whole is designed for the purposes of the Web Application access control. But this approach will give developers virtual team brought troubled team A and team B must be in the Web Application Construction of a unified server, in addition to developers writing in the local-End Application, and then uploaded to the server, and in this version of synchronization between the issue; Also on the remote server because of network security, construction and restrictions, causing other developers to upload, synchronization, testing and other issues.

To solve the above problems, In the

paper, we present a efficiency way by Session combining XML, Web Access Cross-domain applications competence domain / server implementations, website so that all applications can operate their own servers, and no longer need to unify the entire run to the same server, and the whole system is no longer restricted web applications to the same programming language to write.

Keywords: RBAC、Access Control、Cross-Domain

1. 緒論

一個由虛擬團隊合作開發的計畫案，他們總是希望以自身所熟悉的方式建構這個系統。例如團隊 A 習慣使用 JSP 建置 Web Application；團隊 B 習慣使用 ASP.net 建置 Web Application；團隊 C 習慣使用 PHP 建置 Web Application。因此整個計畫案的系統分析師(SA)就必須考量以何種平台作為優先考量，以使其他團隊必須配合以該平台作為開發基礎，在此先假設 SA 優先考量 JSP 的優點而要求其他團隊使用 JSP 開發，且以團隊 A 所架設之伺服器為主要伺服器，其他團隊所架設伺服器為備援之用。由於各團隊所架設伺服器環境不盡相同，在許多的條件限制之下，團隊們會希望採用跨網域的方式提供使用者存取他們所設計的網頁，此時系統分析師(SA)就會考慮使用單一簽入系統(Single Sign-on)且嘗試結合 RBAC 來做到跨網域權限存取控制。

單一簽入系統，目前使用最多的是 Microsoft .net Passport[16,17]機制，以及被廣泛應用的 Kerberos[15,20]機制，還有學者 Liao[8]提出「網際網路單一簽入系統應用」之方法。[5]在此 SA 知道速成的單一簽入系統可以使用 Microsoft .net Passport 所提供的機制，暫不考慮使用 Microsoft .net Passport 可能花費的成本，由於它必需建置在 ASP.net 上，勢必將所有已經開發的 Web Application 從新改用 ASP.net 最為開發基礎。若採用廣泛應用的 Kerberos 機制，則需另外架設驗證主機、發佈憑證、以及未知的開發風險。

本研究使用 RBAC 作為存取控制方式，在使用者登入期間，即產生對應之所屬權限；當使用者需跨越網域或跨越伺服器時，則透過 XML 與會期客戶端(Session Client)使各個獨立伺服器重新取得使用者的存取權限。

2. 文獻探討

2.1 Web Application

Web Application 是一種架設在 Web 伺服器(Server)上，接受使用者資訊、處理並回傳網頁資訊的應用程式，基本的架構是 Client/Server，Server 只負責處理資料，並且和資料儲存區間溝通與交換資料，在處理完成後產生網頁指令，並輸出到用戶端。

用戶端(Client)是一種實作網路通訊的本機應用程式(Local Application)－大部份是瀏覽

器，或是實作網路通訊用戶端的伺服器程式，負責處理/解譯由伺服器傳輸而來的網頁標記資料(tagged data)，然後繪製(rendering)成一個完整的網頁畫面，並且處理存在標記資料中的指令程式碼(scripting language)，讓用戶端具有快速處理與回應的能力。

Web Application 使用的標記資料是一種以標記(tag)定義各項資料的格式，藉以讓用戶端處理的資料流(stream)，網頁使用的標記資料格式稱為 HTML，或者是使用更嚴謹的 XML 格式，裡面包含了文字資料，超連結或是加密的二進位資料(編碼為 Base64 格式的字串)等等。[1]

2.1.1 Session

Session 為使用者於 Web Application 工作階段的期間使用，當用戶進入 Web Application，使用者的工作階段就會開始。當使用者有一段時間沒有作業，或明確的結束工作階段(如「登出」)，工作階段就會結束。當工作階段存在時，它是專屬於各別使用者，每個使用者都有單獨的工作階段。

Session 可視為變數資訊，假設用戶存取某伺服器中的 A Web Application，即 A Web Application 中的作業皆能存取這個資訊，但同樣位於該伺服器中的 B Web Application 是無法存取該資訊。

當工作階段第一次啟用時，Web Application 會建立唯一識別碼 ID，該識別碼會放置於客戶端，而所有的 Session 資訊皆儲存於伺服器記憶體中。[25]

2.1.2 Cookie

Cookie[12,13]是 Web Application 中最常見的一種儲存少量資料的機制，其 Cookie 可分為暫時性 Cookie(Transient Cookie or Session Cookie)和持續性 Cookie(Persistent Cookie)兩種。持續性 Cookie 的限制如下：

1. 只允許每個網站儲存 20 個 Cookie。
2. 有些瀏覽器對 Cookie 數量加上絕對限制，總數不超過 300 個 Cookie。
3. Cookie 內容大小不可超過 4096 個位元組。

暫時性 Cookie 即在使用者與 Web Application 結束工作後消失。[6]

2.1.3 XML

XML 是從 1996 年開始有其雛型，並向 W3C(World Wide Web Consortium)[29]提案，而在 1998 年二月發佈為 W3C 的標準、延伸自 SGML(The Standard Generalized Markup Language)的標記語言，由於具有開放的架構，加上其不被技術平台所限制的特性，在 XML 推出後，隨即廣為業界所接受。XML 的運用廣泛，只要透過標準化的標籤制定方法，系統開發者可利用它來開發企業間通訊的資料格式、甚至是系統通訊協定。[3]

2.2 RBAC(Role-Base Access Control, 以角色為基礎之存取控制)

RBAC[22]主要的觀念是在使用者及資訊資源權限之間加進角色(職務)的概念，先將所需的權限指派給該職務，然後再將使用者指派至所屬的職務，以職務來決定使用者是否有執行某個物件的權限。RBAC 基本模型[22]請參考圖 2-1 所示：

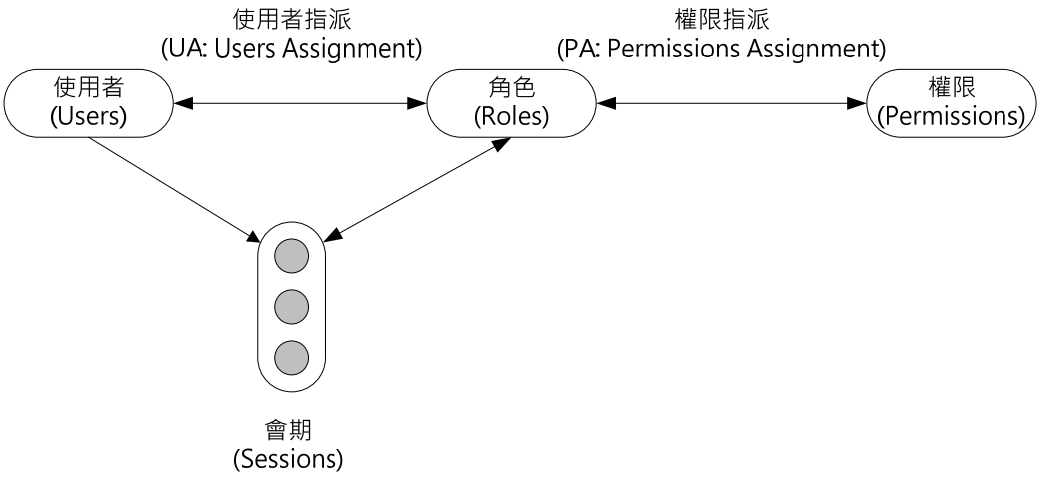

圖 2-1 RBAC 基本模型[22]

3. 流程架構

3.1 系統架構

本研究機制包含兩個組要架構：(1)RBAC 授權存取控制，(2)跨網域處理機制。本章節以「輕度障礙生數位學習平台」建置為例，在這個平台上切分兩個團隊分別製作「概念學習」(位於服務伺服器 A)及「問題導向學習」(位於服務伺服器 B)兩區塊。

使用者欲瀏覽「概念學習」服務，須向服務伺服器 A 請求服務，服務伺服器 A 在確認使用者未曾登入過，將會指示使用者進行登入作業；服務伺服器 A 將使用者所提供之帳戶密碼進行驗證，在驗證成功後，向 RBAC 授權中心請求該使用者的角色以及相依權限，隨後 RBAC 授權中心將該使用者相關權限回傳給服務伺服器 A，再由服務伺服器 A 與使用者建立起 Session List，其中包含了使用者的基本資訊以及完整的可用權限列表，另外配置一組唯一的 Session Client。若權限列表中有包含該使用者可使用「概念學習」，即可作業相關服務。

使用者欲瀏覽「問題導向學習」服務，該項目位於服務伺服器 B，由於 Session List 僅建立於服務伺服器 A 與使用者之間，因此必須另外建立服務伺服器 B 與使用者之間的 Session List。服務伺服器 B 檢知使用者含有一組唯一的 Session Client，即使用該組 Session Client 向服務伺服器 A 取得完整的 Session XML，並且將其 Session XML 重新轉換為服務伺服器 B 與使用者之間的 Session List。本研究機制基礎架構如圖 3-1 所示：

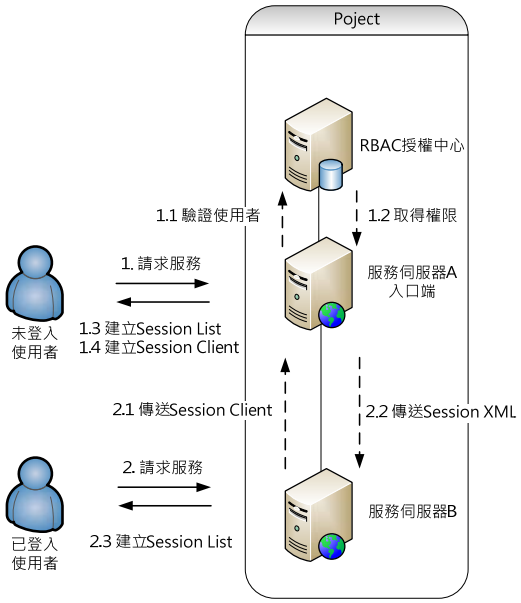

圖 3-1 系統基礎架構

3.2 RBAC 授權存取控制

一個註冊的使用者，經由登入後，必須經由 RBAC 存取控制取得自己的 Session List，首先使用該用戶的 ID，透過資料庫關連表取得 Role ID，由於使用者可能會有一個以上的角色身分，因此在此建構一個 Roles 陣列存放可用的 Role ID，在存取的同時檢測，該使用者是否在允取的時間內以及允許的地點存取，若是拒絕則從陣列表中剃除。該部分可參考局部流程圖 3-2。

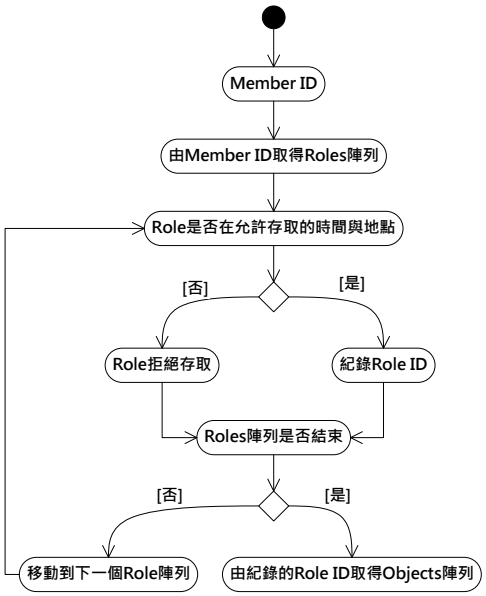

圖 3-2 Role ID 取得流程

由取得的 Role ID 即可使用資料庫關聯取得 Object ID，一個角色可能會關聯許多不同的 Object，因此建構一個 Objects 陣列放置可用的 Object ID，在放置的同時檢測該 Object 是否在可用的時間內即允許的地點方可寫入陣列。該部分可參考局部流程圖 3-3。

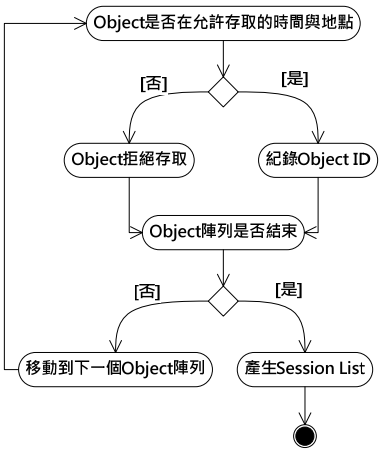

圖 3-3 Object ID 取得流程

在取得完整的可用 Roles 陣列以及 Objects 陣列後，即產生伺服器與使用者之間的 Session List，內容包含使用者相關資訊以及可用的 Role List 和 Object List。該部分可參考表 3-1。

表 3-1 Session List

Session Name	Session Value
ID	yuu
Name	由宇
Email	sfw.sakana@gmail.com
R_Count	2
R_ID_0	browser01
R_ID_1	sysadmin
O_Count	14
O_ID_0	Admin_O2R
O_ID_1	Admin_Objects
O_ID_2	Admin_R2O
O_ID_3	Admin_R2U
O_ID_4	Admin_Roles
O_ID_5	Admin_U2R
O_ID_6	Admin_Users
O_ID_7	Logout
O_ID_8	O_List
O_ID_9	Radmin_EX01
O_ID_10	Session_List
O_ID_11	Session_XML
O_ID_12	Session_XML_Show
O_ID_13	首頁

每個網頁(Web Page)都視為一個 Object，若使用者進入的頁面資訊，並不包含在使用者的 Session List 中，即禁止使用者瀏覽該頁面。該 Session List 僅建立於服務伺服器與使用者之間，在使用者關閉之後或逾時 30 分鐘後即會自行作廢。

3.3 跨網域處理機制

由於 Session 特性，它無法轉交或授予其他的 Web Application，因此跨過網域遞送這些 Session List 則必須透過其他的方法。大部分的開發者首先想到的就是利用 Cookie 來儲存這些 Session List。但大多數瀏覽器對 Cookie 有幾個條件上的限制：

1. 只允許每個網站儲存 20 個 Cookie。
2. 有些瀏覽器對 Cookie 數量加上絕對限制，總數不超過 300 個 Cookie。
3. Cookie 內容大小不可超過 4096 個位元組。

由於整個 Session List，無法預估將來一個使用者會有多少 Roles，以及相依的 Objects 又會有多少個，很顯然的單一網站使用 20 筆是不敷使用的。另外，Cookie

的安全性原則，僅能存取自身網站或是允許的相依網域之下(例如：*.yahoo.com)，當跨越不同的網域名稱時，則無法使用。因此使用 Cookie 來做為跨網域的傳遞媒介就不在此考慮，取而代之的是由服務伺服器建立一組唯一的雜湊碼 Session Client，在產生 Session List 之後一併將 Session Client 遞送給使用者。

當使用者進入另一個服務伺服器時，網頁將會預先檢查使用者對於該服務伺服器是否存在著 Session List，若檢知沒有將會檢測使用者是否含有 Session Client；當服務伺服器取得 Session Client 後，即將 Session Client 遞送回入口伺服器取得 Session XML。Session XML 內容請參閱表格 3-2。

表格 3-2 Session XML

```
<?xml version="1.0" encoding="UTF-8" ?>

<PageAdmin>
      <User ID="yuu">
      <Name>由宇</Name>
      <Email>sfw.sakana@gmail.com</Ema
il>
</User>

<Role Count="2">
      <RID>browser01</RID>
      <RID>sysadmin</RID>
</Role>

<Object Count="14">
      <OID>Admin_O2R</OID>
      <OID>Admin_Objects</OID>
      <OID>Admin_R2O</OID>
      <OID>Admin_R2U</OID>
      <OID>Admin_Roles</OID>
      <OID>Admin_U2R</OID>
      <OID>Admin_Users</OID>
      <OID>Logout</OID>
      <OID>O_List</OID>
      <OID>Radmin_EX01</OID>
      <OID>Session_List</OID>
      <OID>Session_XML</OID>
      <OID>Session_XML_Show</OID>
      <OID>首頁</OID>
      </Object>
</PageAdmin>
```

服務伺服器由入口伺服器取得該使用者的 Session XML，即能重新轉換為該服務伺服器與使用者之間的 Session List。完整流程為圖 3-3。

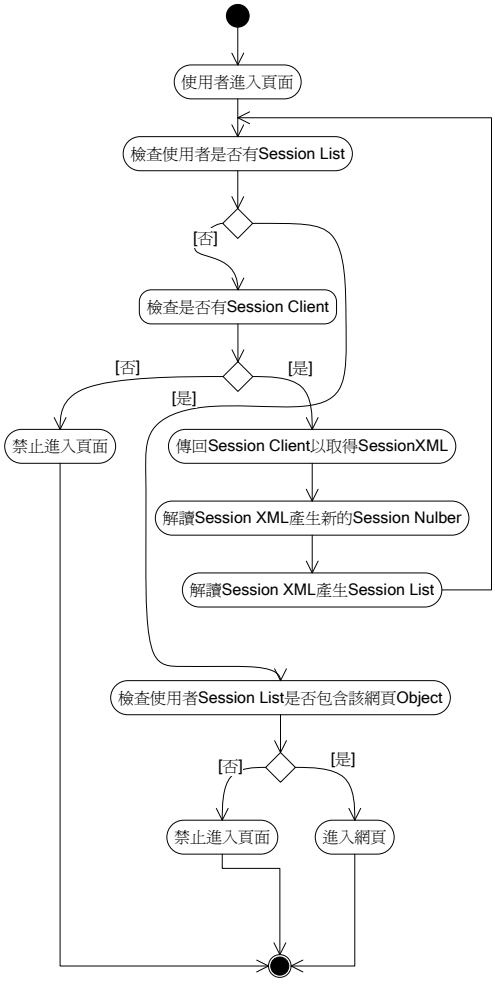

圖 3-3 使用者進入頁面權限檢知

4. 實作設計

根據第三章的實作設計，實做了一個簡易 RBAC 的存取控制系統，包含登入、登出、範例網頁 A(Admin_Users)、範例網頁 B(Radmin_EX01)。新增一個有效帳戶：Demo。給予具有閱覽範例網頁 A 及範例網頁 B 之角色。

在跨網域部分使用兩台不同 IP 之伺服器，分別為 203.72.1.27 與 203.72.1.59，前者為服務伺服器 A 且為入口網站；後者為服務伺服器 B 屬於遠端機台。將範例網

頁 A(Admin_Users)放置於 203.72.1.27 伺服器；範例網頁 B(Radmin_EX01)放置於 203.72.1.59 伺服器中。

使用者使用 Demo 帳戶進入該系統後，將會經由 RBAC 取得該帳戶之角色，且取得該角色可用權限(Web Page)。接著得到一組 Session Client 及 Session List。假設已知下列兩個鏈結點：

- 服務伺服器 A 範例網頁 (Admin_Users)：http://203.72.1.27/PageAdmin_02/pa_Admin/Admin_Users.jsp
- 服務伺服器 B 範例網頁 (Radmin_EX01)：http://203.72.1.59:8080/EX01/pa_Radmin/EX01.jsp

在未授權的情況下瀏覽，即重新導向至未授權訊息頁面。請參閱圖 4-1：

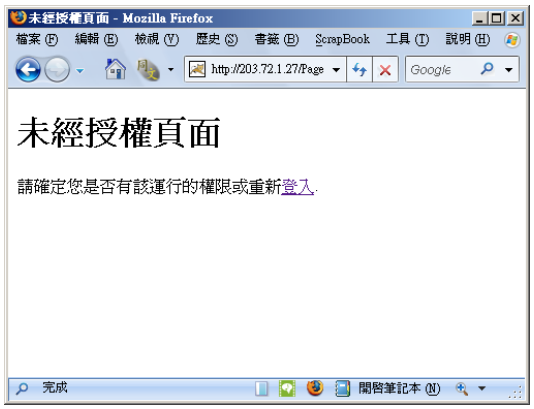

圖 4-1 未經授權頁面

按下頁面上重新登入鏈結即可轉向至登入畫面，在進入登入畫面後，進行帳號密碼驗證，請參閱圖 4-2：

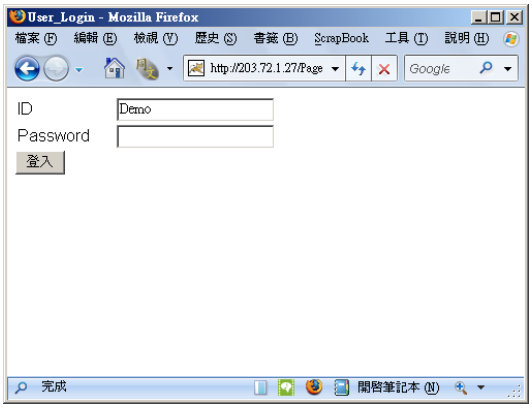

圖 4-2 登入介面

登入成功後，後台介面分為兩部分，左邊為功能清單，其功能清單是經由 RBAC 列表後為可使用的 Object(Web Page)，同時代表著該使用者可使用的權限；右邊為執行功能介面。請參閱圖 4-3：

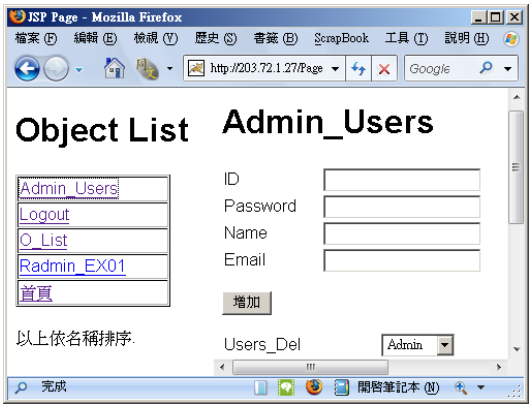

圖 4-3 後台介面功能

從圖 4-3 的 Object List 中可以看到 Demo 用戶可使用的權限(Web Page)包含了：Admin_Users、Logout、O_List、Radmin_EX01、首頁。假設 Demo 用戶得知另一個網頁範例的連結：

- 服務伺服器 A 範例網頁 (Admin_Roles)：http://203.72.1.27/PageAdmin_02/pa_Admin/Admin_Roles.jsp

由於 Demo 用戶的 Session List 中並未包含該權限，亦未出現於於 Object List 介面上，因此會被強制重新導向至未授權訊息頁面，可參閱圖 4-1。

當 Demo 用戶在閱覽 Radmin_EX01，Radmin_EX01 屬於另一遠端伺服器(服務伺服器 B)，在 Demo 用戶登入服務伺服器 A 時，並不會將其用戶之 Session List 主動的帶給服務伺服器 B，因此用戶閱覽時將會主動的提交一份，在用戶登入時所產生的獨立唯一碼 Session Client 給服務伺服器 B，服務伺服器 B 將此唯一碼遞送給入口網站(服務伺服器 A)取得該唯一碼的 Session XML，成功取得該用戶的 Session XML 將會重新建立 Demo 用戶與服務伺服器 B 之間的 Session List。請參閱圖 4-4：

圖 4-4 服務伺服器 B 範例網頁 (Radmin_EX01)

5. 結論

在跨網域網頁應用程式的傳統方式，無非是利用 Cookie 來攜帶資訊，但 Cookie 只能攜帶少量有限的資料，且也只能在設定在指定的網域之下。經由 RBAC 後的權限清單的數量是無法預估的，有限的 Cookie 沒辦法包容下未知的權限清單。或許較小的企畫專案不需要那麼大量資訊，或在有限的情況下可以使用 Cookie 作為資訊傳遞的媒介，但亦會受到域名的規範；當然可以另外花費成本申請一個二級網域，架設網域伺服器配置三級網域來使整個系統可以順利的存取 Cookie，但難免有無法另外消耗成本或認為不應該如此殺雞用牛刀。

現階段所開發的 Single Sign-on 也具備了跨網域的能力，且能整合兩個不同站台的網頁應用程式，但它們都必須經過一個獨立的認證伺服器，且要使用者要認可該憑證，因憑證可分為大型機構所頒布及私人所發布兩種，若是為了節省成本而使用私人所發布之憑證，大部分的瀏覽器會詢問使用者是否接受該憑證以繼續瀏覽網頁，但部分的使用者會誤認為網頁無法預覽，要無痛讓使用者接受憑證，就得另外消耗成本來使用大型機構所頒布之認證。且使用者在登入後，專案內的站台只能得知該使用者是否能使用該服務，無法細部的設置使用者在該服務站台內的權限。

本研究機制提供一簡單跨網域 RBAC 存取控制之解決辦法，可免除架設認證主機、頒布憑證，以及其它為此因應所開發所需之成本。各團隊在網頁應用程式開發上，只需依照自身所習慣方式開發，且建置在自身的網頁伺服器中。用戶由入口網站登入後，透過 RBAC 方式取得完整的使用者權限，在需要跨過各團隊所建置之伺服器時，將由 Session XML 來遞送用戶應有的使用者權限，在重新產生 Session List 之後，就如同用戶在原先伺服器上所登入時所取得之權限。

參考文獻

[1] 朱明中，「Web Application 首部曲：了解 ASP.NET 基礎架構」，http://www.microsoft.com/taiwan/msdn/columns/jhu_ming_jhong/A-ASP.NET_Architecture.htm

[2] 余俊賢，「以 RBAC 為基礎建構網頁存取管控機制」，交通大學資訊管理研究所碩士論文，2002 年。

[3] 李長庚，「一個開放的 Web-Based Sign-On 服務架構」，國立交通大訊管理研究所碩士論文，2002 年。

[4] 林千代，「可攜性 RBAC 資訊系統架構之研究」，朝陽科技大學資訊管理系碩士論文，2003 年。

[5] 林孟勳，「結合 RBAC 授權之網站單一簽入機制研究」，世新大學資訊管理學系碩士論文，2006 年。

[6] 郭貞伶，「網際網路單一簽入系統結

合 RBAC 授權機制之研究」，世新大學資訊管理學系碩士論文，2006 年。

[7] 曾俊豪，「以角色為基礎作網頁伺服器的存取控制之系統設計與實作」，台灣大學資訊工程研究所碩士倫文，2000 年。

[8] 廖英彥，「網際網路單一簽入系統應用」，世新大學資訊管理學系碩士論文，2005 年。

[9] 簡毓麟，「網站單次簽入之企業整合應用」，交通大學電資學院學程碩士論文，2003 年。

[10] 鐘聿光，「RBAC 權限控制系統中登入管理之研究」，中原大學資訊工程學系碩士論文，2006 年。

[11] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn and R. Chandramouli, "Proposed NIST Standard for Role-Based Access Control", ACM Transactions on Information and System Security, Vol. 4, No. 3, pp. 224-274, 2001.

[12] D. Kristol, "Http Cookies：Standards, Privacy, and Politics" , ACM Transactions on Internet Technology, Vol.1, No.2, pp. 151-198, 2001.

[13] HTTP COOKIES Preliminary Specification, http://wp.netscape.com/newsref/std/cookie_spec.html

[14] Java 2 Enterprise Edition, http://java.sun.com/j2ee/

[15] Kerberos, http://web.mit.edu/kerberos/

[16] Microsoft Passport SDK Documents, http://msdn.microsoft.com

[17] Microsoft, "Microsoft .NET Passport Review Guide" http://msdn.microsoft.com/library/default.asp?url=/downloads/list/websrvpass.aspMicrosoft ,2004.

[18] Q. Li, J. Shi and S. Qing, "An Administration Model of DRBAC on the Web", Proceedings of IEEE International Conference on e-Business Engineering, pp.364-367, 2005.

[19] R. S. Sandhu, "Roles Versus Groups", First ACM Role-Based Access Control Workshop, pp. 5-6, 1998.

[20] R. S. Sandhu and P. Samarati, "Accress Control：Principles and Practice", IEEE Communications Magazine, pp.40-48, 1994.

[21] R. S. Sandhu, D. Ferraiolo and R. Kuhn, "The NIST Model for Role-Based Access Control：Towards a Unified Standard", Proceedings of the 5th ACM Workshop on Role-Based Access Control, pp. 26-27,2000.

[22] R. S. Sandhu, E. J. Coyne and C. E. Youman, "Role-Based Access Control Models", IEEE Computer , pp. 38-47,1996.

[23] R. Wonohoesodo and Z. Tari, "A Role Based Access Control for Web Services", Proceedings of IEEE International Conference on Services Computing, pp. 49-56, 2004.

[24] Role-Based Access Control, http://csrc.nist.gov/rbac/

[25] Session, http://livedocs.adobe.com/dreamweaver/8_tw/using/34_obt10.htm

[26] Single Sign-On, http://www.opengroup.org/security/sso/

[27] T. Nykänen, "Secure Cross-Platform Single Sign-On Solution for the World-wide Web", Department of Computer Science and Engineering, Helsinki University of Technology, 2002.

[28] W. B. Shim and S. Park, "Implementing Web Access Control System for the Multiple Web Service in the Same Domain Using RBAC Concept", 8th International Conference on Parallel and Distributed Systems, pp. 768-773, 2001.

[29] W3C, http://www.w3c.org/