可信賴的分散式標準時間服務架構

Trusted Distributed Standard Time Source Architecture

丁培毅
國立台灣海洋大學資訊工程系
pyting@cyber.cs.ntou.edu.tw

徐家暉
國立台灣海洋大學資訊工程系
blackarease@hotmail.com

倪行健
國立台灣海洋大學資訊工程系
enijmax@cyber.cs.ntou.edu.tw

褚芳達
中華電信研究所
cfonda@cht.com.tw

摘要

在電子商務快速成長的今日，建立可信賴的標準時間源是刻不容緩的工作。在 2002 年 Datum 公司提出一種三層式的安全時戳服務系統架構，使用時間校正擴散標準時間到底層伺服器。由於此種架構中需要大量的參考時鐘，維護這麼多時鐘的頻率會造成系統管理上的困難，並且增加系統提供錯誤時間的機率。本文中提出一種階層式的標準時間源架構，透過取消第三層時間應用伺服器的參考時鐘，以降低校正大量時鐘的困難，以及增加稽核時間值的功能性。本系統中使用多重稽核機制防止各個伺服器可能的偽造，確定所提供時間的正確性。

關鍵字：標準時間源、分散式架構、時間校正程序、代理簽章

Abstract

As e-commerce becomes properous nowadays, it is urgent to construct an infrastructure to provide standard time services. On 2002, Datum Corporation has announced a three-level secure time-stamp service architecture, in which the standard reference time is expanded by calibrating the clock on each server. However, since it is very expensive to maintain numerous reference clocks in the system, we propose a modified architecture which decreases the number of reference clocks in the system and uses multiple auditing mechanisms to prevent each server from fraud.

Keywords: Standard Time Source, Distributed Architecture, Time Calibration Process, Proxy Signature

一、 簡介

公開金鑰基礎建設(PKI)大部分的服務都需要時戳服務作為其提供服務的基礎，如憑證授權中心(CA)發出憑證(Certificate)或是公佈憑證撤銷清單(CRL)時，需要附上對應的時戳，確保當時的時間以及文件的完整性。

時戳服務分為相對時間時戳服務和絕對時間時戳服務。絕對時間時戳服務使用精確校正至 BIPM 的標準時間，並且以此判斷不同時戳之間順序，因此所使用的時間源的精準度非常重要，要維護穩定、標準可稽核的時間源的成本也很高。為了要提供標準、可信賴、可稽核，成本較低的時間來源，我們設計並實作一個分散式的標準時間服務架構，提供精確可信賴的時間值給時戳伺服器，作為其提供服務的基礎。

Datum 公司在 2002 年[3]提出一種三層式安全時戳服務架構，透過時間校正機制由最上層的國家標準時間伺服器(NMI)校正第二層伺服器的時鐘，再由第二層伺服器校正底層時戳伺服器的時鐘。由於該架構中每一層的伺服器都維護獨立的時

鐘，因此整個系統成本昂貴，且數量眾多的時鐘提高了校正的成本，也導致稽核該系統發出時間值是否正確的困難。

為了減少系統中需要維護的頻率來源，降低因頻率偏移或是時間校正程序所產生的時間誤差，我們提出另一種可信賴階層式時間源架構，改善 Datum 架構的問題，分析系統的安全性，並且闡述其稽核機制。

第二節中介紹階層式可信賴時間源架構，分別描述各個層級的伺服器功能，實作校正時間的方法，以及其他相關的問題。透過時戳服務的應用來說明架構設計的原因。第三節中介紹建立稽核伺服器的方法與實作機制。第四節中分析各種偽造時間值的方法，確保稽核機制能發揮作用，第五節為結論。

二、 階層式可信賴時間源架構與實作

2.1 系統架構

為了提供精準、可信賴的時間值，系統以國家標準時間作為時間來源，「國家時間與頻率標準實驗室」(National Measurement Institute, NMI)，以圖 1 的階層式架構校正第二層 TMC 時間伺服器的時鐘，第三層應用服務伺服器則由第二層時間伺服器取得時間值，如此可以得到一個分散式的時間服務架構，其提供的時間值可稽核、可追溯到國際間互相承認的標準時間。

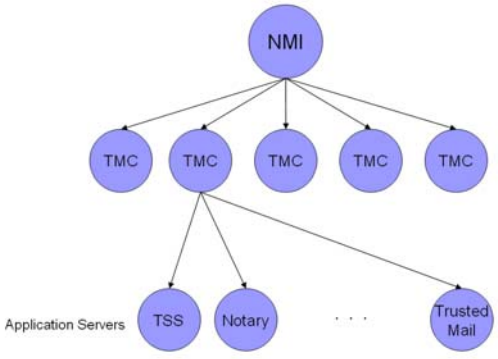

圖 1、階層式時間服務架構圖

NMI 伺服器為系統中標準時間的源頭。其時間卡本身由銫原子鐘提供頻率並同步到國家時頻實驗室的銫原子鐘，時間的誤差控制在 10^{-6} 秒內。該伺服器的硬體放置在國家時頻實驗室內，具有實體安全性，避免硬體遭到破壞或是替換。伺服器上透過電信網路和次級時間源(TMC)伺服器進行時間校正，校正過程的通訊經過雙向認證和加密，以確保傳送時間值之可信賴性與正確性。由於使用電路連線的電信網路，可以避免封包傳遞延遲所造成的誤差，提供精確的校時服務。

TMC 伺服器設置於各地的 PKI 服務業者的機房內，負責提供第三層應用伺服器標準時間。TMC 伺服器使用銣原子鐘或是 GPS 衛星天線來作為參考頻率，並且需要定時向 NMI 伺服器進行時間校正，其時間精準度維持在 10^{-2} 秒內，這個數值對於大部分電子商務的應用已經足夠。TMC 伺服器經由區域網路，透過加密的連線傳送時間值給認證過的第三層應用伺服器，並且將所有提供的時間值記錄下來，應用密碼學的雜湊值(Hash)函式來串連所有的記錄，每隔一段時間公佈記錄的鏈結值於具公信力之媒體上，避免記錄資料遭到人為修改。

第三層應用伺服器提供各種時間相關應用服務，如時戳伺服器，公證伺服器和存證電子郵件伺服器等等。這些伺服器並不維護自己的時間源，而由 TMC 伺服器提供的標準時間作為其服務的基礎。基本上，一台 TMC 伺服器可以透過區域網路服務一定數量的應用伺服器，但是為了平衡 TMC 伺服器的負載量，以及防止 DOS 攻擊，每台應用伺服器必須先向伺服器註冊後才能使用標準時間服務。以時戳服務為例，時戳伺服器收到時戳的請求後，透過認證和加密的區域網路連線向 TMC 伺服器要求標準時間值，對時間值及及相關文件資訊加上代理簽章製作成時戳後，發送給使用者。

在現實世界中，代理簽章是已經運作多年的一種機制。比如說經理可以在休假時，授權給他人代為履行職務，此時職務代理人簽署的文件在授權範圍內就擁有和經理簽署的文件相同的效力。在代理時間結束後，原簽章者就可以收回授權，解除代理人代理簽章的權利和義務。在電子化的文件運作中，我們以基於公開金鑰加密演算法的數位簽章法

來取代傳統簽章的方法。在各階層標準時間服務中，我們也應用密碼學的代理簽章機制，使 NMI 伺服器在嚴密稽核及精確校正後可以授權給每個 TMC 次級時間源伺服器以及時間應用服務伺服器，令其代為對所發行的時間進行簽章，使用者則可以相信時間的正確性可以追溯到 NMI 伺服器。

2.2 系統實作

本系統實作採用 Linux RedHat 9.0，以 OpenSSL 函式庫來實作各種密碼學運算。本章說明架構中如何進行校正時間的通訊以及進行校時的演算法。由於 TMC 伺服器的頻率精準度較差，而且必須追溯校正至國家標準時間，所以在一定時間內，需要向 NMI 服器要求線上校正服務。如圖 2 所示，完整的時間校正流程如下：

1. TMC 伺服器透過公眾電話網路(PSTN)或是專線與 NMI 伺服器建立連線。
2. NMI 伺服器產生一亂數 R，並傳送 R 給 TMC 伺服器。
3. TMC 伺服器使用自己的 RSA 私密金鑰對 R 簽章，並傳送 R 的簽章 Y 給 NMI 伺服器。
4. NMI 伺服器使用 TMC 伺服器的憑證來驗證 Y，若驗證正確，則 TMC 伺服器身份認證成功。
5. TMC 伺服器同樣對 NMI 伺服器進行相同的認證，以確認 NMI 伺服器的身份。
6. 雙向認證成功完成後，NMI 伺服器會先傳送 1 秒後的標準時間值。
7. TMC 伺服器在收到時間值後，等待時間標記訊號(Ontime Mark Signal)。
8. 在 1 秒後，NMI 伺服器傳送時間標記訊號給 TMC 伺服器。
9. TMC 收到後，立刻進行校時演算法進行校正時間。
10. 完成校正通訊。

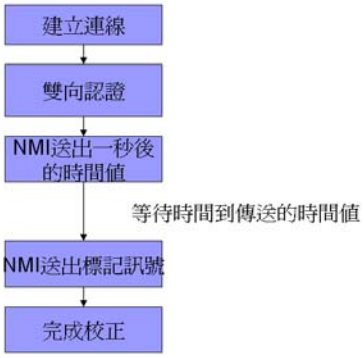

圖 2、校正時間通訊流程

由於 TMC 伺服器必須在時間校正過程中持續提供時間服務，需要滿足時間嚴格遞增以及平順改變的要求，避免時間"回到過去"或是"急速前進"的情況對時間相關應用造成負面的影響，因此我們在校正時間過程時，採用線性的方法來修正時間，以一個固定的比例(亦即下面的 Calibrate_rate)來修正 TMC 伺服器維護的時間源之時間值。以下為校正時間的演算法：

Calibrate_rate = 0.1

Offset = T_{NMI} - T_{TMC}

Total_adj_period = | Offset | / Calibrate_rate

Expect_Time = T_{TMC} + Total_adj_period

$\Delta T = (T - T_{TMC}) * Calibrate\_rate$

If (T < Expect_Time)

    $\Delta T = (T - T_{TMC}) * Calibrate\_rate$

Else

    $\Delta T = Offset$

If (Offset > 0)

    $T' = T + \Delta T$

Else

    $T' = T - \Delta T$

上式中，T_{TMC} 是得到標準時間值時的 TMC 伺服器本身時間、T_{NMI} 則是當時 NMI 伺服器的標準時間值、Offset = T_{NMI} － T_{TMC}，表示總共需要調整的時間差值，Offset 為正的話表示 TMC 時間較 NMI 時間慢；而為負的話表示 TMC 時間較 NMI 時間快。Expect_Time = T_{TMC}+ | Offset | / Calibrate_rate 即為預定 TMC 時間調整完成的時間。在 T_{TMC} 到 Expect_Time 之間的時間值都是經

由驅動程式依照 Calibrate_rate 來對時間值作增加或是減少，因此在這段時間中，TMC 伺服器提供的時間值 T'並非標準的時間值，在 Expect_Time 以後，TMC 伺服器提供的時間值才會是校正正確的時間值。為了讓 TMC 伺服器的管理者可以依照應用的需求來調整 Expect_Time 的時間，管理者可以經由設定校正比例值來加快或是減慢時間校正的速率。其值會介於 0~1 之間，用於表示每秒最多可以修正的秒數。愈接近 1 的校正比例，其達到 Expect_Time 的時間會較短；反之則需要更長的時間才能校正完成。

透過完整的雙向認證機制以及加密的通訊，我們可以確保惡意的使用者沒有辦法在進行校正通訊時欺騙 TMC 或是應用伺服器，並且也無法得知送出的標準時間值和其他資訊。使用線性時間校正的演算法，可以讓 TMC 伺服器不會因為進行校正時間而停止提供時間值服務，並且能提供嚴格遞增的時間值。

2.3 與 Datum 安全時戳服務架構的比較

Datum 公司的時戳服務架構在每一個伺服器中，都維護一個時間經由上層伺服器透過網路校正下層伺服器的時間，一旦校正完成後，下層伺服器的時鐘即依靠本身的頻率來源來運轉，若頻率來源無法達到所需要的精確度時，時間值的誤差就會變大。由於應用伺服器數量眾多，因此系統中需要維護相同數量的時間源，造成系統建構成本極高，在實際運作時，系統管理者必須要定時調校數量龐大的參考原子鐘的頻率，否則頻率一旦有誤差，系統提供的時間值就不能保證其精準度。

為了避免維護過多的頻率源，本文提出的架構中，應用伺服器並不維護時間源，而是由 TMC 伺服器來提供時間值給應用伺服器使用。由於 TMC 的數量遠少於應用伺服器的數量，我們只需要維護 TMC 伺服器的參考頻率來源的精確，以及定時校正 TMC 伺服器的時間源，即可確保 TMC 所提供的時間值精準度在一定範圍內。此外，由於 TMC 伺服器不用為應用伺服器校時，減少校時的程序，亦降低時間稽核的複雜度。TMC 伺服器和應用伺服器之間透過網路來傳送時間值，為了避免偽造，兩者通訊之前必須通過雙向認證，確認對方的身份。

三、 階層式架構稽核機制

上述分散式時間源主要透過固定期的時間校正將國家標準時間延伸到 TMC 伺服器上，然而其可信賴性必須透過嚴密的稽核機制來建立，以下將說明如何應用線性連結機制支援標準時間源架構的名種稽核程序。

稽核的概念就是將一份資料記錄下多份不易修改的拷貝，再將這些拷貝保存在不同利益取向的人員手中，以確保沒有人能篡改所有拷貝的資料。

3.1 線性鏈結機制

Stuart Haber 等人針對單一時戳伺服器，發展一種基於鏈結確保時戳記錄無法更改的方法[4]。這個方法主要應用抗碰撞的雜湊函式完成。以下我們應用線性鏈結機制和數位簽章來稽核 TMC 伺服器和應用伺服器。

以時戳伺服器(TSS)的運作為例，時戳服務中心必須記錄每一筆簽發的時戳，如圖 3，並且為這些記錄建立線性鏈結，每一段時間將鏈結值公佈在具有公信力且無法仿冒變造的媒體上，使得時戳服務中心無法任意更改、替換與安插時戳簽發記錄，藉此向時戳服務使用者證明自己公正地提供時戳服務，其程序如下：TSS 伺服器簽發完成第 n 筆時戳時，計算鏈結值 $L_n = h(T\_SN_{n-1}, MD_{n-1}, SN_{n-1}||T_{n-1}, TL_{n-1}, L_{n-1})$，其中包括時戳序號 T_SN_{n-1}、文件的訊息彙記 MD_{n-1}、TMC 伺服器發出的時間序號和時間資訊 $SN_{n-1}||T_{n-1}$、TMC 伺服器的第 n-1 次鏈結值 TL_{n-1} 與時戳伺服器上一次發出的鏈結值 L_{n-1}，其中 h 為單向抗碰撞雜湊函式。計算出 L_n 後，將 T_SN_n、$SN_n||T_n$、MD_n、TL_n，以及 L_n 寫入記錄檔中。時戳伺服器所簽發的第 n 筆時戳 $TS_n = (T\_SN_n, MD_n, SN_n||T_n, TL_n, L_n, \sigma_n)$，其中 σ_n 代表時戳伺服器為電子資料 $(T_SN_n||MD_n||SN_n||T_n||TL_n||L_n)$所產生的數位簽章。

所有產生出來的鏈結值 L_n 在固定時間內都會公佈在時戳服務伺服器上，讓使用者可以驗證時戳伺服器運作的公正性。若時戳伺服器已經公開了 L_i 和 L_j 這兩個鏈結值，則任何人都沒有辦法更改第 i 筆到第 j 筆時戳之間的任何記錄而不被發現。

將線性鏈結機制也應用於次級時間源(TMC 伺服器)，防止任何人修改其記錄資訊。TMC 伺服器在發出第 n 筆時間值時，計算 $TL_n = h(MD_{n-1}, SN_{n-1}||T_{n-1}, TL_{n-1})$，記錄文件的訊息彙記 MD_{n-1}，時間序號和時間資訊 $SN_{n-1}||T_{n-1}$ ，和鏈結值 TL_{n-1}，並公佈於 TMC 伺服器上。TMC 伺服器所發出的時間值格式為$(SN_n||T_n||TL_n)$，第三層應用伺服器(如時戳伺服器)需要將這些資訊包在時戳內送給使用者，並且保存這些資訊於本身的稽核檔案中，以供正確性稽核之用。與 TSS 伺服器相同，TMC 伺服器也必須定時公開其 TL_n 於具公信力之媒體上，防止 TMC 伺服器在發出時間值後，更改其時間記錄檔中各項欄位。如圖 3 所示，若 TMC 伺服器公佈了鏈結值 TL_2 和 TL_7，則任何人都沒有辦法更改第 3 筆到第 6 筆時間值的任何一欄位而不被發現。例如：使用 T_4'來替換第 4 筆時間值中的 T_4 的話，則必須重新計算 $TL_5' = h(MD_4, SN_4'||T_4', TL_4)$，基於單向抗碰撞雜湊函式的特性，對於一已知的 TL_5，無法在合理時間內找到 $SN_4' || T_4'$，使得 $TL_5' = TL_5$，稽核者可以由 TL_2 持續驗證至另一已公佈的鏈結值 TL_7，此時會發現重新計算的鏈結值與公佈的鏈結值至少有一個地方不同。

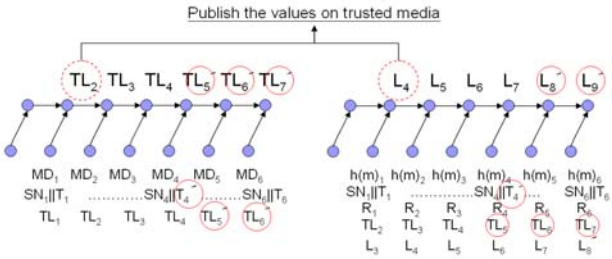

圖 3、TMC 和 TSS 的鏈結值

若 TMC 只公開了 TL_2，TMC 伺服器更改第 3 筆時間值中的 T_3，則可以重新計算 $TL_4' = h(MD_3, SN_3'||T_3', TL_3)$、$TL_5'$、$TL_6'$…，並且更改公佈的鏈結值為 TL_4'、TL_5'、TL_6'，當持有 TL_3 使用者會發現公佈的 TL_3'與自己原先取得的時間鏈結不相符，基於數位簽章之不可否認性，使用者即可舉發 TMC 伺服更改了時間值簽發記錄。

3.3 系統稽核機制實作

TMC 和 TSS 伺服器在六個小時必須公佈記錄檔和鏈結值於伺服器上，供 NMI 和 TMC 稽核員進行驗證，稽核員依照驗證的結果，來決定伺服器是否有偽造的可能，並停止提供時間值給有偽造可能的伺服器。系統中包括五種稽核機制：

1. 伺服器稽核員：

在 NMI 和 TMC 伺服器上，必須定時查核所服務的下一層伺服器是否正常運作，透過比對記錄檔內容，確保記錄值是一致的。以 NMI 稽核員為例，TMC 伺服器在向 NMI 伺服器進行校正後，必須要在記錄檔中新增一筆 $SN \| T \| N\_SN_n \| TL_n$ 記錄，其中 N_SN_n 為 NMI 伺服器上校正時間的序號值，取代原本記錄文件訊息彙記的欄位。而在 NMI 伺服器中，每次校正完成必須新增一筆 $N\_SN_n \| T_{NMI} \| NL_n$ 校正時間記錄，其中 T_{NMI} 為校正時間所送出的時間值，NL_n 為記錄檔的鏈結值。建立記錄檔後，未來可以透過比較 NMI 和 TMC 的記錄檔來驗證 TMC 伺服器是否有依照 NMI 伺服器提供的時間來校正本身的時間值，並且誠實地提供時間給 TSS 伺服器。

2. 自動偵測機制

TSS 伺服器和 TSS 的客戶端，透過網路校時協定(NTP)校正本機時間值，可以將本機的時間值誤差維持在正負數秒內。使用本機時間來檢查伺服器處理要求所花的時間是否合理。以 TSS 伺服器為例，在 TSS 系統時間 T_0 向 TMC 伺服器要求時間值，T_1 收到 TMC 伺服器送來的時間值 T。TSS 伺服器在向 TMC 伺服器要求時間值時，必須依照本機的時間，計算 T_0 到 T_1 之間所花費的時間。若回應所花費的時間太長，表示 TMC 伺服器可能在這段期間內進行偽造時間值。得到時間值 T 後，TSS 伺服器必須檢查 T 的時間值是否落在合

理的範圍內。在考量網路傳輸以及系統時間誤差的狀況下，我們設定 T 的合理範圍應介於 T_0-ΔT 秒到 T_1+ΔT 秒之間，若 TMC 伺服器提供超過合理範圍內的時間值，TSS 伺服器可以發現 TMC 伺服器的時間值有偽造的可能，自動送出警示給 NMI 伺服器，如果 TSS 伺服器沒有和 TMC 伺服器勾結串通的話，此自動偵測機制可以發揮相當的作用。

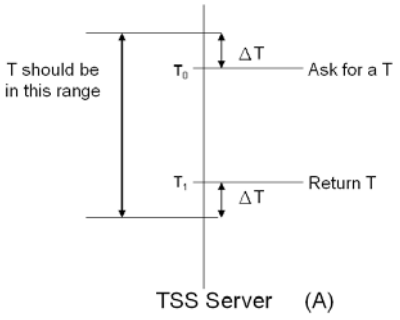

圖 4、自動偵測機制

3. 驗證時戳的客戶端：

使用者向 TSS 伺服器要求時戳後，將文件和其時戳送給另一位使用者，藉此向其證明文件的時間，為了和要求時戳的使用者區分，這裡我們稱呼其為驗證者。驗證者得到文件和其時戳後，必定會先驗證時戳的簽章，若驗證者不相信發出時戳的 TSS 伺服器，驗證者可以透過 TSS 客戶端軟體檢驗 TSS 伺服器的公開記錄檔中各個欄位以及鏈結值的正確性。若使用者與 TSS 伺服器串謀，則驗證者有機會經由比對 TSS 記錄檔和時戳的內容來得知。

下一節中，應用以上三種稽核機制，來分析時間源系統中可能的攻擊和偽造。

四、 時間源攻擊偽造分析

在電子商務應用中，時間值是很關鍵的資訊，操弄時間可能獲取的利益讓時間相關應用服務伺服器可能遭受到賄賂或是駭客攻擊，目的在偽造有利的時間值。因此在這個系統中必須防止提供時間值的次級時間源 TMC 伺服器與第三層應用伺服器偽造時間值。

4.1 時戳伺服器(TSS)可能的偽造問題

4.1.1 時戳要求流程

如圖 5、使用者要求時戳的流程如下，將想要求時戳文件 m 的訊息彙記 h(m)傳送給 TSS 伺服器，TSS 伺服器將收到的文件訊息彙記送給 TMC 伺服器作為要求時間值的憑證。TMC 伺服器收到訊息彙記後，便會立刻送出當時的時間序號 SN、時間值 T、以及記錄檔鏈結值 TL 給 TSS 伺服器，並且記錄發出的 SN||T 以及文件訊息彙記 h(m)於記錄檔中。TSS 伺服器收到 SN||T 和鏈結值 TL 後， 製作時戳 TS = (SN || T || h(m) || TL || L || W_m)，其中 W_m 為代理簽章授權書，TSS 伺服器使用代理簽章的私密金鑰 Privkey 對 TS 進行簽章， 得到 σ = SignPrivkey(SN || T || h(m) || TL || L || W_m)，將 TS 和 σ 傳送給使用者，完成時戳要求。

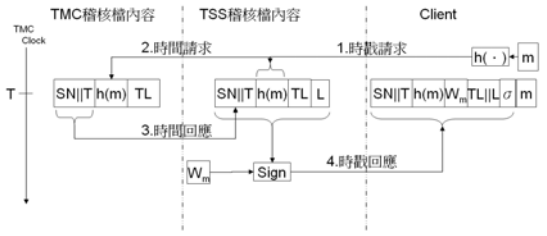

圖 5、時戳要求流程

4.1.2 修改由 TMC 伺服器取得的標準時間來發時戳

若使用者和 TSS 伺服器串謀修改時戳中的時間值，TSS 伺服器可以依照使用者要求來修改得到的標準時間值，再製成時戳送給使用者。基於伺服器的效率考量，TMC 伺服器對於所發出的時間不加以簽章而只是在記錄檔中登錄下來。如圖 6，TSS 伺服器在取得 TMC 伺服器提供的標準時間值 T 後，若任意更改時間值為 T'後再製成時戳送給使用者。使用者收到時戳後，由於時戳中的時間值是由 TSS 伺服器和使用者串謀偽造的，故使用者並不會去驗證時戳中的時間值。但是收到時戳的驗證客戶端軟體在驗證時戳內資訊與 TSS 伺服器公開資訊是否符合時，可以偵測出來 TSS 伺服器是否有進

行偽造。

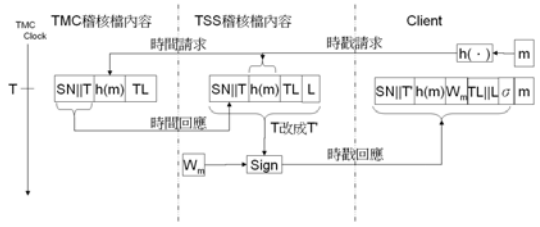

圖 6、TSS 送出修改後的時戳

考慮第二種狀況：圖 7 中 TSS 伺服器把修改過的 T'值也記錄在時戳稽核記錄檔中，由於時戳中的 T'值與 TSS 伺服器稽核檔內的 T'值是相同的，使用者客戶端軟體無法發現 TSS 伺服器作假，但是因為 TSS 伺服器稽核檔內的 T'與 TMC 伺服器稽核檔內的 T 值不一致，TMC 或是 NMI 伺服器的稽核員可以在固定時間的稽核流程中發現 TSS 伺服器作假，TSS 伺服器就無法再得到標準時間的服務了。

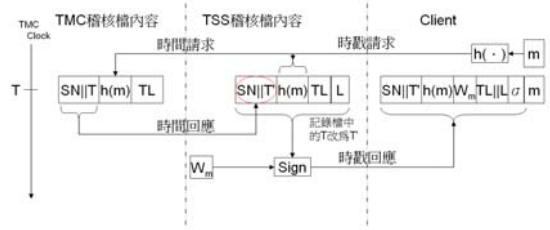

圖 7、同時修改 TSS 稽核檔中的時間值記錄

考慮第三種狀況：圖 8 中假設 TSS 伺服器對調了兩個時間值 T_0 與 T_1，製成兩個時戳分別送給使用者 u_0 和 u_1，TSS 伺服器為了逃避 TMC 及 NMI 伺服器定時的稽核，在 TSS 稽核檔案中應該還是留下正確的記錄，假設 $h(m_0)\neq h(m_1)$，此時有兩種可能發現問題的方法：第一，由於 TSS 伺服器必須在收到 $SN_1||T_1$ 以後才能夠對 $h(m_0)$作出第一個時戳，如果 T_1 與 T_0 間隔很大，使用者 u_0 的客戶端軟體在發現 TSS 伺服器有相當的延遲時應該要主動查詢 TSS 或是 TMC 的稽核記錄，會發現在稽核記錄中 $h(m_0)$對應的時間值為 T_0，並非所收到的 T_1。第二是所有收到時戳的人如果不信任 TSS 伺服器時也都可以運用驗證客戶端軟體對時戳內容進行查驗，此時在 TSS 或是 TMC 的稽核記錄中就會發現 $h(m_1)$所對應的時間值為 T_1，並非時戳內所記錄的 T_0。

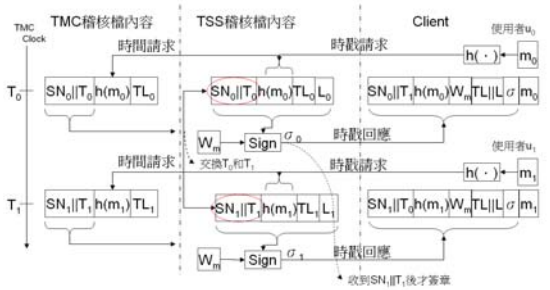

圖 8、調換時戳中標準時間值的順序

4.1.3 送出過多的時間值要求

在這個分散式時間源架構中，TSS 伺服器和 TMC 伺服器在同一個區域網路中且互相信賴。使用者如果只收買某一 TSS 伺服器，所需花費的成本當然比串通 TMC 伺服器低。由於 TSS 伺服器必須擁有文件的訊息彙記才能向 TMC 伺服器要求時間，如圖 9 所示，如果 TSS 伺服器能夠為多個均勻分佈的 h(m')值向 TMC 伺服器預先取得對應的時間值，TSS 伺服器可以運用生日攻擊法(Birthday Attack)和使用者共同產生文件 m，使其和先前某一個文件 m_i 具有相同的訊息彙記 $h(m)=h(m_i)$，此時 TSS 伺服器不向 TMC 伺服器索取時間值，而是直接發給先前取得的 $SN_i||T_i,h(m_i),TL_i,L_i$ 所製作成的時戳，使用者可以得到較早之時間值 T_i。這樣子的作假模式沒有辦法透過 NMI 伺服器或是 TMC 伺服器的稽核機制篩檢出來，只能透過加長 hash 函式的位元數來降低生日攻擊法的威脅性，目前使用的 160 bit SHA1 基本上可以符合要求，如果需要的話也可以使用 256 或是 384 bit 的 SHA 演算法。另一種避免此種問題的方法是要求 TSS 伺服器在稽核檔內記錄使用者相關資訊，例如：假設服務的使用者是採用會員制的話，可以記錄其會員號碼，如此在稽核時可以追蹤到使用者端的時戳，當 TSS 伺服器向 TMC 伺服器取得很多時間值卻沒有發出時戳時，TMC 的稽核程序就可以查覺異常狀況了。再一種考量由於 TSS 伺服器需要對時戳簽章，每秒

鐘最多能發出的時戳數會有上限，雖然 TSS 伺服器可以先準備好時戳相關資料，等待較空閒時再一一簽章，然而由於代理簽章委託授權書期限有限制，未來的時戳請求數量不可預期，TSS 伺服器基本上不該接受過多的時戳請求，因此 TMC 伺服器對於某一 TSS 伺服器的時間請求應該有所限制，如此也限制了此種作假方式中假時戳記錄的個數。

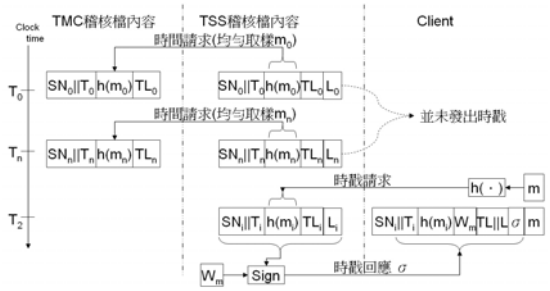

圖 9、送出多個時間值但未發出時戳

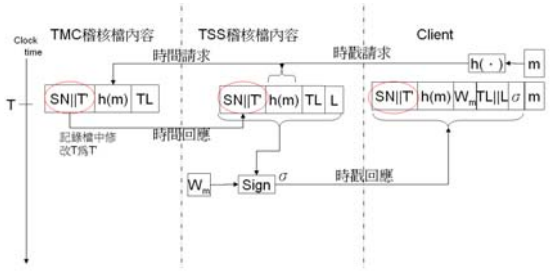

圖 10、無法透過鏈結值驗證 TMC 伺服器偽造時間值

4.2 TMC 伺服器可能的偽造方法

TMC 伺服器負責提供標準時間值給第三層應用伺服器，在整個信賴時間源機制中位於中階，一旦使用者對於某一 TMC 伺服器失去信任，將會影響其下所有的第三層應用伺服器，如果 NMI 伺服器的稽核機制發覺 TMC 伺服器的公正性或是正確性有問題的話，也會停止代理簽章的授權，影響到其下所有的應用伺服器。

如圖 10，TMC 伺服器提供錯誤的時間值 T' 給 TSS 伺服器，並加入錯誤的時間值到記錄檔中，TSS 伺服器依照得到的時間值 T'，製作成時戳送給使用者，並將錯誤的時間值加入本身的記錄檔中。由於 TMC 記錄檔內容和 TSS 記錄檔內容都是完全一致的，因此不論是沒有標準時間參考源的使用者或驗證者，或是擁有標準時間的 NMI 和 TMC 的稽核員都無法透過驗證記錄檔和鏈結值來得知 TMC 伺服器偽造時間。在現有機制下，TMC 伺服器仍有一點空間可以作假，TMC 偽造時間的範圍受到發出時間值的密集度限制，如果在 TMC 伺服器上有兩筆連續時間記錄 T_1、T_2，TMC 伺服器更改 T_2 時，最多只能將 T_2 改為一個大於 T_1 的值，如果 TMC 發出的時間值非常密集的話，它可以偽造的空間就很小，因此需要適當的機制來維持 TMC 伺服器的最低工作量。一種方法是當 TSS 伺服器閒置的時候，利用自己最新的鏈結值作為文件，每隔一固定時間向 TMC 伺服器要求時間值，得到的時間值並不發行時戳，只記錄在記錄檔中。這樣可以保證 TMC 伺服器中兩筆連續時間請求的間隔在一定的範圍內，同時也限制了 TMC 伺服器偽造時間值的範圍。

4.3 TMC 伺服器和 TSS 伺服器共謀

TMC 伺服器若是和 TSS 伺服器共謀更改已發出的時間值記錄檔以及鏈結值，以避免 NMI 稽核員對記錄檔稽核時發現 TMC 和 TSS 伺服器的記錄檔不一致。如圖 11 所示，假設 TMC 伺服器目前只提供時間給一台 TSS 伺服器，並且只公佈了 TMC 伺服器上的鏈結值 TL_2 和 TSS 伺服器上的鏈結值 L_4，它們欲共謀更改時間值 T_4 為 T_4'，TMC 伺服器需要根據 T_4'重新計算鏈結值 TL_5、TL_6 和 TL_7，成為 TL_5'，TL_6'和 TL_7'，TSS 伺服器需要修改對應的三筆資料，並重新計算鏈結值 L_7，L_8 和 L_9，成為 L_7'，L_8'和 L_9'，只要 T_4'沒有早於 T_3 或是晚於 T_5，NMI 稽核員無法稽核出 TMC 和 TSS 共謀偽造的問題。由於 TSS 在 T_4 之後，已發出兩筆時戳，其中的時間值為 T_5 和 T_6，鏈結值為 $TL_6||L_7$ 和 $TL_7||L_8$，故只要其中一個時戳的驗證者對 TSS 伺服器進行驗證，就可以發現 TSS 伺服器修改過記錄檔，鏈結值分別為 $TL_6'||L_7'$和 $TL_7'||L_8'$，即可向 NMI 稽核員檢舉 TSS 伺服器偽造，而 NMI 稽核員即可比對兩者的記錄檔發現 TSS 和 TMC 共謀。

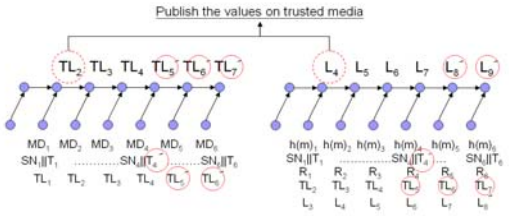

圖 11、TMC 和 TSS 共謀修改時間值

五、 結論

本論文中提出的階層式標準時間源架構成本比 Datum 公司在 2002 年提出的標準時戳服務架構低且可信賴度高。由於我們取消應用伺服器所使用的頻率源，故只需要維護及校正架構中 TMC 伺服器所使用的頻率來源，可以減少架構中校正時間的次數以及校正參考頻率源的次數，降低因為不精準的頻率來源所造成的時間值錯誤的機會。

而為了確保 TMC 伺服器上的時間值精確度，TMC 伺服器必須定時向 NMI 伺服器進行校正，以取得 NMI 的標準時間值來校正 TMC 伺服器的時間值。本論文實作透過電話線路校正時間的程序，並使用線性校正時間的方法，確保時間值可以平順地改變，維持 TMC 伺服器的時間值嚴格遞增的特性，並且可以在校正時同步提供標準時間服務，使服務不需因為校正時間而中斷。

除了維持系統所提供時間的精確度，伺服器的公正性也是使用者是否相信服務的因素之一。本論文中提出數種稽核的機制，來增加各個伺服器偽造或是修改時間值的難度，使用者亦可透過稽核機制來驗證伺服器的記錄，驗證得到的時戳內容是正確無誤的，這個階層式的標準時間源架構衡量使用者的方便性，與對時間值準確性的一般需求，運用較可信賴的兩層式架構提供標準時間的服務。

六、 參考文獻

[1] D. W. Jones, “Auditing Elections, “ in *Communications of the ACM*, vol. 47, C. o. t. ACM, Ed., 2004, pp. 46-50.

[2] I. Tsukasa, K. Noriyuki, I.Michito, I. Kuniyasu, K. Noboru, G. Tadahiro, S. Tomonari, and M.Takao, “Development of the Trusted Time Stamping System,” *Journal of the National Institute of Information and Communications Technology*, 2003.

[3] Datum, “Trusted Time commercial white paper,” Datum, 2002.

[4] S. Haber and W. S. Stornetta, “How to Time-Stamp a Digital Document,” *Journal of Cryptology*, 1991.