國立交通大學

資訊科學與工程研究所

碩 士 論 文

安全資料庫之高使用度網頁使用者介面設計與

儲存資料完整性檢測實作

Secure Database Design and Implementation for

Integrity Verification and High Usability User Interface

研 究 生：李柏青

指導教授：曾文貴　教授

中 華 民 國　103 年 9 月

安全資料庫之高使用度網頁使用者介面設計與

儲存資料完整性檢測實作

Secure Database Design and Implementation for Integrity Verification and High Usability User Interface

研 究 生：李柏青　　Student：Po-Ching Li

指導教授：曾文貴 教授　　Advisor：Dr. Wen-Guey Tzeng

國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

September 2014

Hsinchu, Taiwan, Republic of China

中華民國 103 年 9 月

安全資料庫之
高使用度網頁使用者介面設計與
儲存資料完整性檢測實作

學生：李柏青　　指導教授：曾文貴 教授

國立交通大學資訊科學與工程研究所 碩士班

摘要

醫療資料電子化行之有年。在現今的社會中，人們對於醫療資源的需求不再局限於單一區域；如何整合、管理並且適當地利用區域之間的醫療資料，使病患可以獲得最恰當的醫療服務是一個重要的議題。隨著雲端科技的成熟，將醫療資料集中於雲端儲存裝置統一管理，不僅節省管理的成本，也可以便利區域間對於醫療資料的交流與分享。但是，醫療資料含有病患的個人資料、生理資訊，需要極度重視其私密性；在此前提下，為了滿足醫療資料庫的需求，雲端服務系統也必須提供常見的資料庫操作。除此之外，為了維護醫療服務的品質，使用者需要特別注意醫療資料的正確性；在使用資料之前，應該確認雲端服務系統提供廠商有正確儲存、回傳無誤的資料。

針對上述議題，我們提出一個醫療資料庫系統的雛型-SDEMR (Secure Distributed EMR)，以 MIT-CSAIL 團隊開發的 CryptDB 元件[1]為基礎，採用洋蔥加密法確保儲存醫療資料的私密性，同時支援常見資料庫的操作，如：新增、刪除、修改、排序、整數相加、比對查詢等。而在此系統之中，我們提供了資料完整性檢測的機制，將 PDP(provable data possession)完整性檢測[2]系統修改、融入我們的系統；此機制可檢測儲存在雲端的醫療資料是否正確無誤，為醫師看診時所參考的醫療資料，多做一層把關。另外，為了可以讓使用者更直覺地操作本系統，我們參照高使用度使用者介面的設計原則、方法[3]，設計網頁使用者介面取代指令命令模式，降低使用者操作的門檻。我們也參照行政院衛生福利部推動的電子病歷管理系統中的單張基本格式[4]，建立表單、生成醫療資料，用以模擬我們建置的系統功能。

關鍵字：完整性檢測，加密資料庫，電子醫療記錄。

Secure Database Design and Implementation for Integrity Verification and High Usability User Interface

Student：Po-Ching Li　　Advisors：Dr. Wen-Guey Tzeng

Institute of Computer Science and Engineering
National Chiao Tung University

Abstract

Electronic Medical Records (EMRs) have been used for many years. In today's medical service, patients use the service in different areas. How to integrate, manage and exchange the EMRs in different areas is very important. We can apply the issue to cloud storage service to ease the cost of EMRs management. In this way, it becomes easier to share the EMRs. On the other hand, we should concern the confidentiality and database operations of the EMRs. Besides, the cloud storage system should confirm the EMR's correctness so that the doctor can trust the EMRs and treat patients well.

In our research, we design a medical database prototype-SDEMR (Secure Distributed EMR). We use CryptDB[1], an encrypted database developed by MIT-CSAIL team as our basic component. It applies onion encryption to confirm the data stored on it and supports common MySQL operations such as EQUAL SELECT, ORDER, ADD etc. In our system, we apply PDP(provable data possession) protocol[2] to verify integrity of EMRs stored in our system.

Besides, to make a friendly interface, we design the user interface by referencing the principles of high usability of a user interface [3]. We also make a mini medical database to simulate our system. We followed the suggested standards from the EMRs Standard Management System of the Ministry of Health and Welfare of Executive Yuan to build the records, tables etc.[4]

Keywords：Integrity Verification, Encrypted Database, EMR.

誌　　謝

光陰荏苒，碩士兩年的學涯一轉眼就到了終點。這兩年受實驗室的栽培不論是在精進實作技術上、思考還有解決問題的能力上、團隊的溝通合作上都有很大的成長。這兩年，遭遇許多困難、挑戰，也曾經迷惘過，慶幸生活周遭的許多貴人，聽我分享、和我討論、給我建議，提供我繼續前進的動力。

首先，我想要感謝的是我的指導教授：曾文貴老師，在第一年讓我們很自由地摸索研究的方向、精進自己的專業技術、練習報告的技巧，並指導、確認我們的進步。第二年是最充實的一年，每週固定的討論，提示我們研究的方向，給予種種的建議，讓我們對於自己的研究不會毫無頭緒又不失自己闖蕩的空間。除此之外，老師也一再提醒我們對於做研究該有的執著，每次都讓我印象深刻、受益良多。另外，十分感謝能夠蒞臨並且給予指導的口試委員：蔡錫鈞老師和孫宏民老師，兩位老師提出的問題，讓我體會到自己對於研究、表達能力和對於問題的理解上還有很多地方需要加強，是我必須再精進的方向，謝謝兩位老師不吝指導。

其他還有實驗室的學長們，在我們尋求協助的時候，總是熱心地給予最大的支持；也時常詢問我們有沒有研究上的問題，對於我在研究的路途上，你們是我最大的後盾，永遠的王牌。也要感謝過去學姐研究的成果，幫助我在自己的研究上，更有方向可循。還有實驗室的學弟們，平常沒什麼機會照顧你們，但是需要麻煩你們的時候，總是義不容辭，出手相救。

接著是我的同學們，大家一起交流、想問題、拼通霄的時光是這兩年我最難忘的。很高興有你們三個，讓我的研究生活，不孤單、更有動力。再來是我的朋友們，實驗室以外的生活是精神的糧食，幸虧有你們才能夠讓我在遇到瓶頸、思考被制約的時候喘一口氣，感謝你們這段時間的陪伴、包容！

最後是我的媽媽、爸爸和姐姐，提供我所需的所有生活資源，相信我在研究上面的努力，時常關心我的生活多於研究的進度，謝謝你們的支持，我才能夠無憂無慮的完成這兩年的研究。

兩年是短暫的，但是回憶雋永。因緣際會下可以認識這麼多人，受到大家的幫助，和大家一起成長是很開心的事情。祝福大家在各自的路上都有更好的發展，也希望這兩年的情誼能夠持續。

李柏青　謹誌

103 年 9 月

目錄

表目錄

圖目錄

第 1 章 引言

1.1 前言

台灣自民國 84 年開始實施全民健康保險，累計看診資料以及病歷記錄。大量的醫療記錄，使得紙本病歷記錄不便於管理與儲存的問題浮上檯面；而隨著資訊技術的進步，病歷記錄的電子化管理開始普及。在民國 92 年，台灣完成健保卡的 IC 卡化[5]，病患在看診時，只需透過健保 IC 卡和醫療機關提供的讀卡機，就醫記錄便會直接上傳至中央雲端儲存系統集中儲存。

而為了使病患得到最準確的診斷、最有效的醫療服務，各醫療機關之間提供相關醫療資料交流日趨頻繁。如下圖一：病患 P 習慣到醫院 A 看診，醫院 A 擁有 P 病患長期的醫療病歷記錄。某次外地旅遊的突發事件，迫使 P 病患必須到醫院 B 看診，為了提供更好的醫療服務，醫院 B 向醫院 A 發出要求，希望可以索取病患 P 過去的醫療病歷記錄。在此情境之下，利用雲端儲存服務裝置來做醫療資料的整合與管理，不僅可以降低各醫療機關自力維護資料庫的成本，也能夠更有效地滿足分享、交流醫療資料的需求。而大量病歷記錄的累計對於醫療機關而言是很重要的資產，不僅有利於幫助病患做到更準確的診斷，而大量的病歷資料更可以提供醫學研究所需的數據統計、觀察，以利醫學技術的進展。

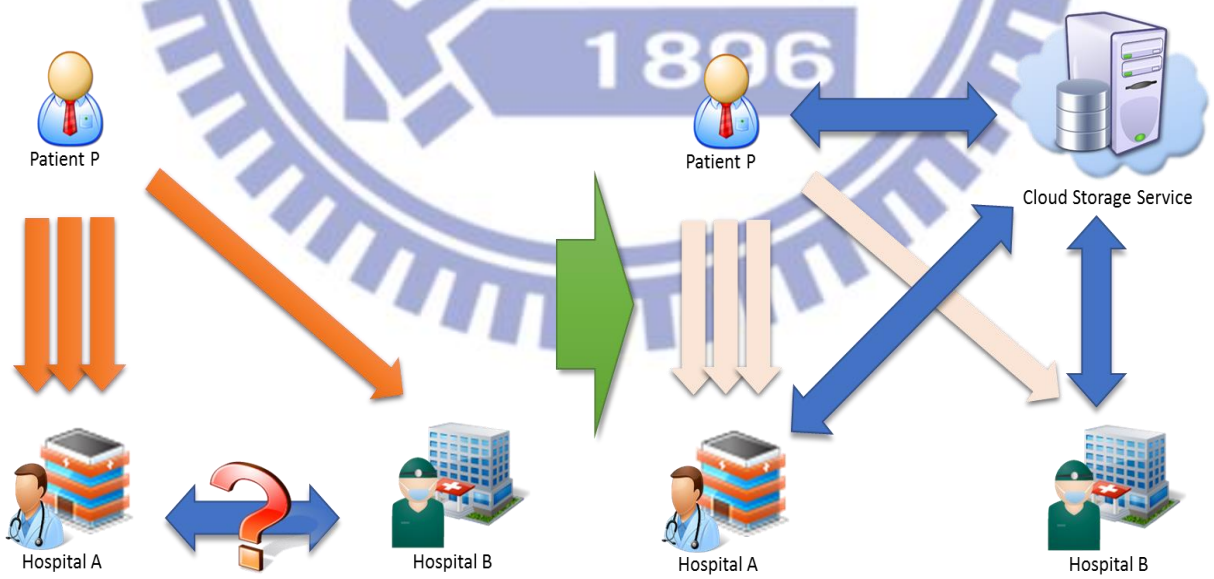


圖 一、跨區域醫療服務示意圖

我們希望可以將醫療資料置於雲端儲存服務裝置統一管理以建立雲端醫療資料庫。但是，醫療資料屬於非常敏感的資料，為了維護病患的隱私，如何確保資料的私密性是一個很基本且重要的需求。同時，為了能夠有效地利用整合的醫療資料來提供病患更完善的服務，雲端服務系統也應該要能夠提供新增、刪除、修改、排序、整數相加、比對查詢...等常見的資料庫功能。另外，醫療機關或診間醫師只有在具備相關身份或是在病患同意授權的情形下，才能夠進行醫療資料的存取，是故，雲端服務系統必須具備存取控制的機制。現實生活中，不肖雲端儲存服務裝置的提供廠商可能會刪除較少使用的資料以節省儲存成本;外部惡意攻擊者可能會嘗試侵入醫療資料庫,並進一步地修改資料。兩者在醫療單位以及使用者未察覺資料錯誤的情形下，可能造成診斷上的誤判、危害病患的權益甚至是生命安全，因此，如何確保資料的完整性也是一個非常重要的課題。是故，我們認為一個雲端儲存服務裝置上的醫療資料庫應該具備以下功能：

- 確保電子醫療資料的私密性(confidentiality)。
- 支援對醫療資料進行新增、刪除、修改、排序、整數相加、比對查詢...等常見的資料庫功能。
- 對醫療資料良好的存取控制。
- 醫療資料的完整性檢測(integrity verification)。

1.2 研究目標

保障儲存資料的私密性是雲端儲存服務的基本需求，Raluca Ada Popa 等學者[1][6][7]設計洋蔥加密法，並實作加密關聯式資料庫－CryptDB。CryptDB 利用加密技術保障資料的私密性;並且利用洋蔥加密法,使 CryptDB 可以對密文資料進行新增、刪除、修改、排序、整數相加、比對查詢等操作。但 CryptDB 沒有支援儲存資料的存取控制和完整性檢測的功能。

我們認為：醫療資料的完整性可以保障醫事人員及病患在讀取資料時的正確性，避免造成資料上的誤判。為了利用 CryptDB 原本具備的保障資料私密性、支援一般 MySQL 常見操作的特性，並且使其符合醫療資料庫使用需求，我們加入完整性檢測機制於 CryptDB。

我們將完整性檢測的計算加入於洋蔥加密法的機制中，在插入資料時，醫療資料被加密後，我們對密文計算完整性的資訊並儲存。在查詢資料的時候，解密以前，我們先驗證密文資料的完整性，若通過才解密。

1.3 貢獻

我們提供了一套具有密文資料完整性確認的安全醫療資料庫應用系統—SDEMR(Secure Distributed EMR)。我們的系統開發是基於由 MIT 的 CSAIL 團隊所開發的安全資料庫系統 - CryptDB，並進行資料完整性機制的整合。CryptDB 能夠支援在資料為加密的狀態下，針對 int 及 text 資料型態的資料進行新增、刪除、修改、排序、整數相加、比對查詢...等常見的資料庫功能操作。也支援在對資料進行存取以及操作時，仍然確保操作資料的私密性，不被雲端服務提供廠商、外來惡意攻擊者所監聽或竊取。但是，在 CryptDB 資料庫系統的設計之中，並無法提供資料的完整性確認機制。為了補足此項不足之處，我們分別引進了完整性確認(integrity check)機制來與 CryptDB 進行整合，以達到本研究目的。

- 使用完整性確認的方法來對醫療資料庫進行資料完整性確認：系統會在解密存取的資料之前，先針對資料進行完整性的檢驗，若通過檢驗才進行解密。而此機制可確保每次看診前，醫師連線至醫療資料庫取得的病歷資料皆是正確無誤的、沒有遭到惡意竄改，為醫師參考的病歷資料多做一層把關。

我們參考行政院衛生福利部的電子病歷管理系統[4]，設計小型的醫療資料庫模型，並且生成資料以供驗證上述的功能。我們也遵照高使用度網頁設計原則[3]，提供網頁使用者介面，取代指令命令模式，降低本系統使用者的操作門檻。

1.4 全文架構

我們的論文共有五個章節，以下將分別介紹第二章：相關研究，探討目前學者對於加密資料庫—CryptDB、完整性檢測以及使用者介面設計的研究。第三章：介紹我們系統中的基礎元件：加密資料庫—CryptDB，解釋洋蔥加密法的基本概念，並提供資料來源，給予有興趣的讀者了解更細部的技術。第四章：詳細介紹我們的系統使用的完整性檢測系統—PDP(provable data possession)，以及系統的相關變形。第五章：系統設計與實作，從巨觀到細部地介紹我們的醫療資料庫加密系統與完整性檢測的架構設計，實作技術的摘要；並且展示實作成果，包含：資料完整性檢測的實驗、使用者介面的展示和解釋實驗資料的參考依據。第六章：結果與討論，以我們的系統現況為基礎，討論未來發展的方向。

第 2 章 相關研究

2.1 CryptDB 相關

一般的資料庫安全，是透過使用者存取控制來限制資料庫內資料的存取，只有經過授權的使用者可以透過資料庫存取操作被授權使用的資料。然而在對於一些敏感的隱私資料的保護，如醫療資料等，只有使用者存取控制是遠遠不足的；需要更進一步的考慮到當資料庫的伺服器被入侵或者是被管理資料庫的管理員刺探時儲存資料的防護。因此，MIT 的 CSAIL 團隊提出了一個具有加密存取操作的關聯式資料庫系統，CryptDB[1][6][7]。CryptDB 可以針對加密的資料進行不同的查詢操作，如使用確定式的(deterministic)加密演算法所加密的資料欄位可以支援等式及關鍵字比對的查詢，利用次序保存式的(order preserving)加密演算法所加密的資料則可以進行比較查詢，而利用同態(homomorphic)加密演算法進行加密的數值資料則可以進行相加的操作。除此之外，CryptDB 可以根據資料型態以及資料的安全需求來制定加密的安全層級，整數型態的資料庫欄位分成三種加密：比對、次序、同態相加。其中，每一種加密都有數層的加密法，是為洋蔥加密法(onion encryption)。每個層的加密方法可以達到不同的目的，例如：使用確定性演算法加密的資料類型屬於較低安全度的，其會洩漏任兩個加密過的資料是否相等的資料；而使用次序保留加密演算法的資料型態則會進一步洩漏加密資料的次序資訊；為要求最高的私密性，資料用上述的加密法加密得到的密文，會再使用隨機(random)加密演算法來進行加密，此類型的資料無法進行任何查詢的動作。CryptDB 可以透過洋蔥式的加密機制使得加密資料可以支援各種不同的查詢操作，並且保證當資料庫系統所洩漏的任何資料不會危害到資料本身的私密性。

另外，CSAIL 團隊也致力於 CryptDB 上關鍵技術更進一步的研究，例如：保有明文次序關係的加密方法(Order-Preserving Encoding)。他們提出一套保有明文次序的加密系統[22]：在伺服器端，利用二元搜尋樹(Binary Search Tree)的概念來記錄密文之間的次序關係；使用者在新增和查詢資料時，必須透過和伺服器的互動，來協助伺服器找到這份密文在這棵二元搜尋樹的位置。這套系統可以達到 IND-OCPA(indistinguishability under ordered chosen-plaintext attack)等級的安全度，意即系統運作的過程中，伺服器不會得到明文關於次序以外的任何資訊。

2.2 完整性檢測相關

資料的完整性議題是探討資料是否真的被完整的保存在雲端中。完整性的探討是希望系統提供使用者一個稽核的方式以確保資料有被正確的儲存，但使用者稽核的時候手邊並沒有所儲存的資料內容。圖二呈現一個使用者儲存資料之後，再進行完整性檢測的範例示意圖。範例中使用者是使用一種簽名演算法，對資料區塊進行簽章，使得檢測時，儲存系統可以回傳指定的資料區塊與相對的簽章，接著使用者就可以進行檢驗。

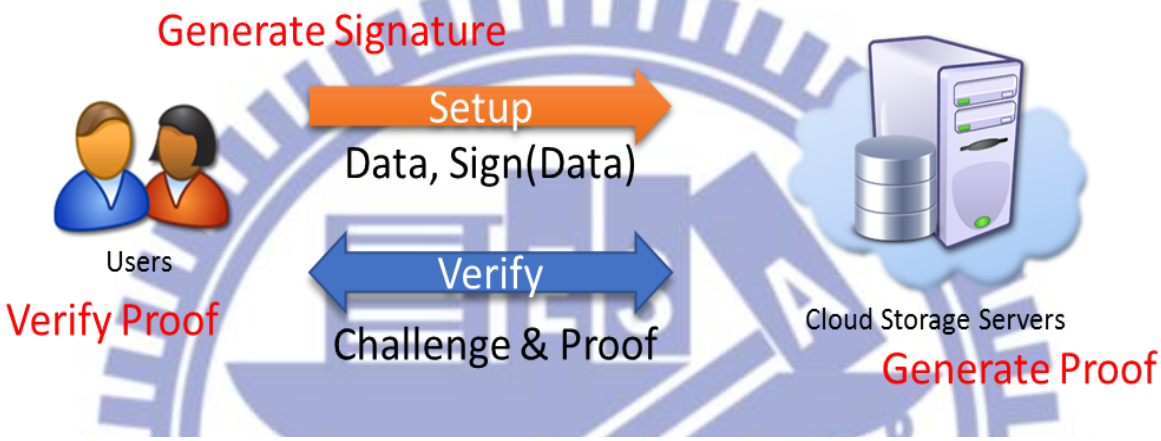


圖 二、完整性檢測示意圖

參考現有完整性檢測相關的文獻，有許多完整性檢測協定已經被提出，其中 Giuseppe Ateniese 等學者提出的 PDP (Provable Data Possession)[2]和 Kevin D. Bowers 等學者提出的 PoR (Proofs Of Retrievability)[8]為當前完整性檢測的兩大概念。PDP 強調如果系統沒有完整儲存資料，則無法通過使用者的驗證程序。PoR 則是強調當資料儲存到雲端之後，系統必須要能夠向使用者証明資料可以被取回，並且在數學定義上說明，整個證明的過程可以將儲存的資料萃取出來。這兩大概念都有接續的研究，特別是針對傳輸資料量的改善，與增加支援動態的完整性檢測。C. Chris Erway 等學者提出以 skip list 為基礎的動態完整性檢測協定，包含資料插入、更新、刪除等運算[9]。Feifei Liu 等學者提出比較有效的動態更新資料完整性檢查協定，改善[9]將計算及傳輸複雜度降低到常數時間[10]。

但上述的方法，直接擴充 PDP 來達到支援動態的資料完整性檢測，都必須犧牲一點安全性。Qian Wang 等學者提出利用 Merkle tree 的資料結構來進行資料的完整性檢測[11]。這個系統中，一份資料的檔案會被切成 n 個區塊，資料庫中除了儲存 n 個資料區塊，還要儲存它們的 hash 值以及上層的 hash 值(共 n-1 個額外的 hash 值)。完整性檢測

的過程中，系統傳送檢查資料的 hash 值、相鄰兄弟節點(sibling node)的 hash 值以及此節點至根節點(root)路徑上的 hash 值給使用者；使用者可以利用這些資訊計算出根節點的 hash 值，並和自己保留的值作比較，以驗證完整性。這個系統可以在不犧牲安全性的情況下，支援動態資料完整性檢測操作。但需要額外維護 Merkle tree，傳輸成本也會增加。當儲存裝置的資料規模龐大，除了儲存成本增加以外，計算及傳輸成本也會非常可觀。

除了讓使用者可以稽核資料是否被正確儲存之外，有學者指出讓使用者授權第三方來進行完整性檢測也是可行的辦法[13]，希望可以減輕使用者檢測完整性的負擔；在使用者在儲存資料之後，交由第三方定期檢驗資料儲存的完整性。可公開檢驗(public verifiability)與私密檢驗(private verifiability)便形成兩個不同的類別。其中也不乏許多醫療病歷資料的應用，Ximeng Liu 等學者便設計系統，醫療資料的擁有者可以指定第三方，代為完成醫療資料完整性檢驗[14]。

2.3 使用者介面相關

資訊科技蓬勃發展，人機介面(HCI, Human Computer Interface)、使用者的操作介面(UI, User Interface)也漸受重視。1970-1980 年間首先提出對於人機介面優劣的評估概念：使用度(usability)，並定義指標：好用、好學(easy to use and easy to learn)。隨後，國際標準化組織(ISO, International Organization for Standardization)於 1997 年也對於使用度定義效率度(efficiency)、有效度(effectiveness)、滿意度(satisfaction)三大概念[15]。

- 效率度：使用者完成要求任務所需要的時間。
- 有效度：使用者對於要求任務的完成度。
- 滿意度：使用者的回饋（和使用者過往的經驗做比較）。

儘管使用度涵蓋三個面向，但早期多數的學者只將研究著重在效率度和有效度，滿意度往往只是附加於前面兩者的成果[16]。關於評估使用者滿意度的研究文獻，在 2002 年 Gitte Lindgaard 和 Cathy Dudek 兩位學者設計實驗評估使用者的滿意度，並且結論影響滿意度的最大原因之一是：網頁給使用者的第一印象，而非過往著重的效率度和有效度[17]。例如：外觀、架構、網頁之間的連結符合使用者經驗中的同性質網頁。除此之外，第一印象具有擴散效應，意即會引導使用者的情緒發展方向，對滿意度造成更大的影響。

1998 年也出現評估網頁使用度的服務：WAMMI (Web Site Analysis Measurement Inventory)[18]，利用涵蓋五個面向的二十個問題記錄使用者的反應，作為網頁使用度的評比，並且和資料庫中累計的大量網頁作比較，得到使用度的排名。另外，也有學者 Nielsen Norman Group 提出：一個使用者介面的開發，五個受測者便可以發現大約 85% 的使用度問題。提出提升使用度的基本方法：每次測試以三到五人為限，觀察受測者的操作反應並且記錄他們的回饋，根據回饋修改介面，再做測試[3][19][20]。

第 3 章 加密資料庫－CryptDB

CryptDB 是一套由 MIT-CSAIL 團隊開發的加密資料庫系統雛型，特點在於：能夠對密文進行資料庫常見的操作，例如：新增、刪除、修改、排序、整數相加、比對查詢...等。確保操作的同時，後端儲存裝置無法藉機竊聽資料，以確保資料的私密性。

3.1 CryptDB 架構

CryptDB 系統主要由兩個元件組成：代理伺服器(proxy)和後端儲存裝置(DBMS, MySQL)。系統環境假設：代理伺服器是可以被完全信任的；而後端儲存裝置則是只能夠被半信任，意即後端儲存裝置會遵照指令行事，但是也會在執行和運算的過程中企圖取得更多的資訊。如下圖二所示，CryptDB 系統主要保證對於後端儲存裝置的資料安全防護，不論是好奇、惡意的儲存裝置提供廠商或是外部的攻擊者針對儲存裝置的滲透，都沒有辦法竊取後端儲存裝置上的資料的資訊。

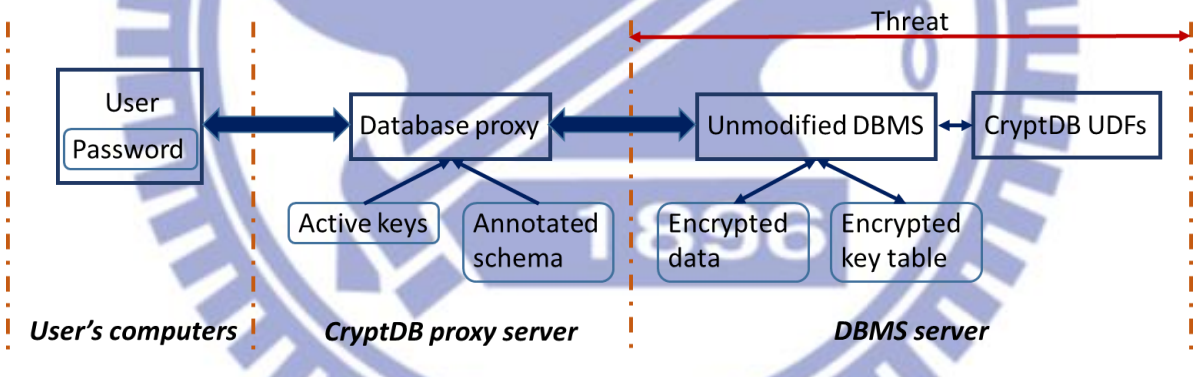


圖 三、CryptDB－系統架構

CryptDB 系統由代理伺服器負責大部分的運算，將收到的 MySQL 指令改寫成後端儲存裝置能夠接收的格式：

- 替換表單名稱至對應的匿名。
- 把值加密成密文。
- 記錄後端儲存裝置表單的結構和加密情形。
- 解密後端儲存裝置回傳的值。

因此，加解密金鑰也是由代理伺服器負責管理，根據相關的表單（table）、行（column）、洋蔥加密法的形式（onion）、層（layer）和代理伺服器中存有的主金鑰（master key）利用隨機排序函式（pseudo random permutation function）計算得到加解密運算的金鑰(1)。

$$\mathbf{K_{t,c,o,l}} = \boldsymbol{PRP_{MK}(table\ t, column\ c, onion\ o, layer\ l)} \quad (1)$$

對於洩漏密文的資訊無虞的小部份運算由後端儲存裝置負責，利用 MySQL 支援的使用者自定義函式(UDF, User Define Function)做洋蔥加密法最外層(RND 層)的解密和執行密文部分運算，如：密文形式的同態相加(homomorphic addition)。

整套 CryptDB 系統編譯時可以直接套用 MySQL 的原始碼，不需要變動。這樣的設計提升了系統安裝的相容性以及除去了 MySQL 資料庫操作指令上的銜接問題。

3.2 CryptDB 核心技術

CryptDB 支援兩種資料型態：int 和 text。一筆 int 型態的資料會加密成三種密文形式：分別是支援比對的相等形式(Onion Eq)、支援排序的排序形式(Onion Ord）及支援相加的加法形式(Onion Add)。相同地，text 型態的資料會加密成三種密文形式 Onion Eq、Onion Ord 和 Onion Search，一樣支援比對和排序，但是在最後一項形式不同，是支援搜尋的搜尋形式(Onion Search)。

每一種密文形式，都是一種洋蔥加密，利用不同的加密方法層層包起。如：相等形式(Onion Eq)第一層會先將明文以支援 JOIN 方法的加密法(JOIN, equality join)加密，再以決定性的(DET, deterministic)加密法包上第二層，最後則是利用非決定性的(RND, non-deterministic)加密法做為最外層的加密。內層的加密法的密文仍然保有部份明文資訊，以利系統對密文進行資料庫操作，如：相同的明文使用決定性加密法加密會得到相同的密文，是故我們可以利用這個特性對決定性加密法的密文進行比對查詢的操作。而最外層的加密法，安全性最高，透露的資訊最少，無法進行任何的密文運算。洋蔥加密法的概念如下圖四所示：

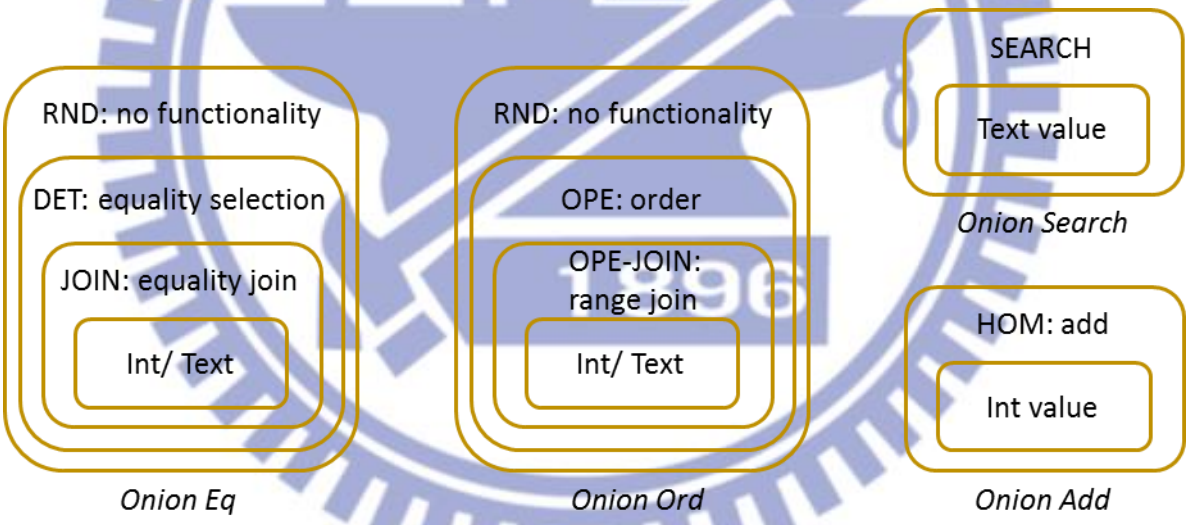


圖 四、CryptDB－洋蔥加密法

3.3 CryptDB 運作流程

當 CryptDB 系統收到使用者的資料庫操作指令(query)時，代理伺服器(proxy)會先對指令進行分析、改寫：

1) 將指令中的表單(table)名稱、行(column)名稱置換成對應的匿名。
2) 利用(1)計算金鑰並加密指令中的常數值(constant)。
3) 根據指令以及表單加密狀況判斷，是否需要要求後端儲存裝置(DBMS)利用使用者自定義函式(UDF)調整洋蔥加密法中的層(layer)。若需要進行調整，則利用 UPDATE 指令要求後端儲存裝置調整洋蔥加密法的密文結構。

完成上述步驟，代理伺服器完成操作指令的改寫，開始對指令以及後端儲存裝置目前的表單狀況進行操作：

4) 判斷是否需要再要求後端儲存裝置的使用者自定義函式進行運算，例如：同態加法(homomorphic addition)。並利用 UPDATE 指令指示後段儲存裝置進行。
5) 後端儲存裝置回傳資料庫操作結果的密文。
6) 代理伺服器收到密文、解密後，回傳至前端呈現給使用者。

以下圖五資料庫表單 ***test1*** 內容為例，使用者執行資料庫操作：

SELECT ***Id*** FROM ***test1*** WHERE ***Name***='***Alice***';

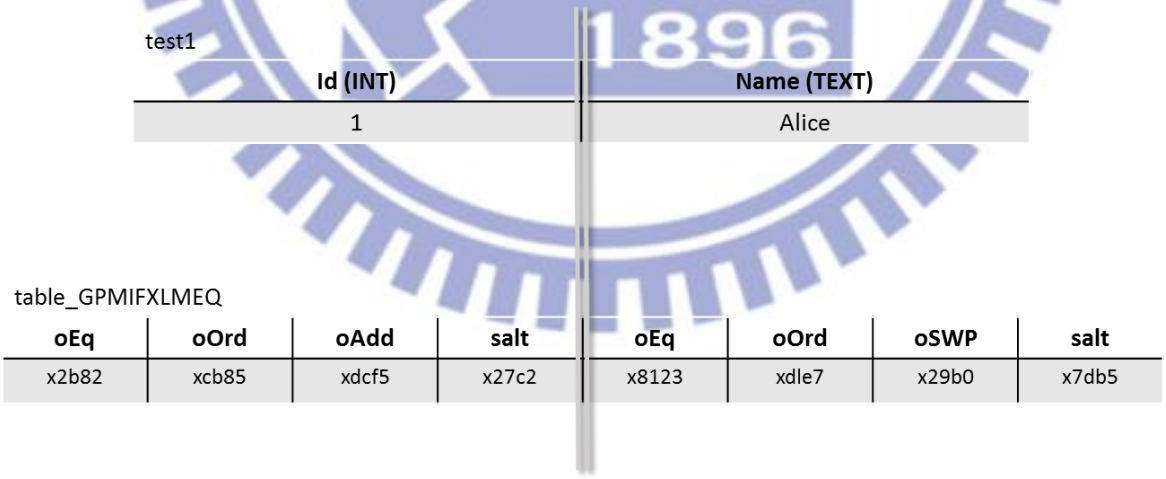


圖 五、test1 表單和 CryptDB 系統中密文表單對應情形

CryptDB 系統收到操作指令：

SELECT ***Id*** FROM ***test1*** WHERE ***Name***='***Alice***';

1) **代理伺服器**將指令改寫，得到改寫後指令：

SELECT ***AGTDoEq*** FROM ***table_GPMIFXLMEQ*** WHERE ***IDNGJoEq***='***Alice***';

2) **代理伺服器**加密常數值 ***Alice*** 成 ***IDNGJoEq***(原 column Name)的 DET 層密文 ***x7213***：

SELECT ***AGTDoEq*** FROM ***table_GPMIFXLMEQ*** WHERE ***IDNGJoEq***=' ***7213***';

3) **代理伺服器**要求**後端儲存裝置**利用使用者自定義函式剝除 ***IDNGJoEq*** 行的 RND 層加密：

UPDATE ***table_GPMIFXLMEQ*** SET ***IDNGJoEq*** =

DECRYPT_RND($K_{T1,C2,Eq,RND}$ ***, C2-Eq, C2-salt)***;

其中，T1: *table_GPMIFXLMEQ*(原 table test1) ,
C2: *IDNGJoEq*(原 column Name),
C2-Eq: *IDNGJoEq* 行的密文,
C2-salt: *IDNGJoEq* 的 salt.

4) **代理伺服器**判斷不需要後端儲存裝置額外的運算。

5) **後端儲存裝置**將 ***IDNGJoEq*** 行中符合比對的 ***AGETDoEq***(原 column Id)欄位值：***x2b82*** 傳回。

6) **代理伺服器**將收到的密文利用相關金鑰

$K_{T1,C1,Eq,RND}$, $K_{T1,C1,Eq,DET}$, $K_{T1,C1,Eq,JOIN}$

解密，

$$DECRYPT\_RND(K_{T1,C1,Eq,RND},\ C1\text{-}Eq,\ C1\text{-}salt);$$

其中，C1: AGETDoEq(原 column Id).

得到 ***1*** 回傳給使用者。

3.4 CryptDB 原始碼(source code)版本簡介

MIT-CSAIL 團隊於 2011 年發佈 CryptDB 論文及程式碼。第一版的 CryptDB 原始碼按論文中所提[1]，分為 Single-Principal Mode、Multi-Principal Mode 兩個模式，命名的差別主要在於加密金鑰的取得方式：

- Single-Principal Mode：加密金鑰源於代理伺服器上的一把主金鑰(master key)，按相關的表單(table)、行(column)、洋蔥加密法的形式(onion)、層(layer)，利用隨機排序函式（pseudo random permutation function）計算得到。
- Multi-Principal Mode：根據使用者的登入密碼建立金鑰鍊(key chain)，並利用額外的表單記錄使用者的對應代號、資料授權情形。不同的資料會依據擁有者的對應代號，用不同的金鑰加密。使用者之間可以透過授權來分享資料。

除了金鑰的生成，兩者的應用方式也不同，相較於 Single-Principal Mode 只能防護後端儲存裝置的威脅；Multi-Principal Mode 達到更高的安全性，能夠防護後端儲存裝置至代理伺服器的威脅。只是 Multi-Principal Mode 加解密時必須採用金鑰鍊的方法拿到金鑰，效率過低。因此 Multi-Principal Mode 在第一次的更新(2013/09/01)便被 CSAIL 團隊移除，直到他們找到更有效率的實作方法。這次的更新也提出新的方法解決舊版本中，Single-Principal Mode 在代理伺服器斷線之後，會失去表單和表單匿名連結的問題；在 CryptDB 資料夾下，新增一個資料夾 shadow，記錄資料庫、表單和其匿名的連結。

第二次的更新(2014/02/07)中，CryptDB 提供呈現、調整資料庫中表單欄位狀況的指令，使用者可以透過指令調整指定欄位的層或是設定指令欄位最低限度的層，以設定這個欄位資料採用的安全程度。

除了這兩次重大的更新，CryptDB 也發佈過幾次修補(patch)，修復記憶體的問題(memory leak)、自動將空欄位填入 NULL 值以提高系統的穩定性。

我們的系統建置在 CryptDB 的 Single-Principal Mode，故在接下來的介紹中將不會再對 Multi-Principal Mode 多加著墨。另外，我們採用的 CryptDB 元件是第二次更新(2014/02/07)後的版本。

第 4 章 完整性檢測－PDP

PDP(Provable Data Possession)完整性檢測協定的主要概念在於：雲端儲存裝置若沒有完整儲存資料，則沒有辦法通過系統的驗證。只要雲端儲存裝置有基本的運算能力，使用者不需要取回原始資料就能檢查資料的完整性，可以有效降低資料傳輸的成本。這個特點使得在雲端儲存裝置上驗證大量資料的完整性更為可行。

下圖六為 PDP 完整性檢測系統示意圖：

(a) 部份顯示使用者上傳資料至雲端儲存裝置前，先計算資料檔案(F)的完整性標籤，將完整性標籤和資料檔案一併上傳(F')；並自己儲存相關的驗證金鑰資訊(m)。

(b) 部份則是闡述，當使用者希望雲端儲存裝置證明資料正確儲存時，會產生挑戰(R)給雲端儲存裝置；雲端儲存裝置利用收到的挑戰和正確儲存的資料檔案計算證明碼(P)，回傳給使用者驗證。

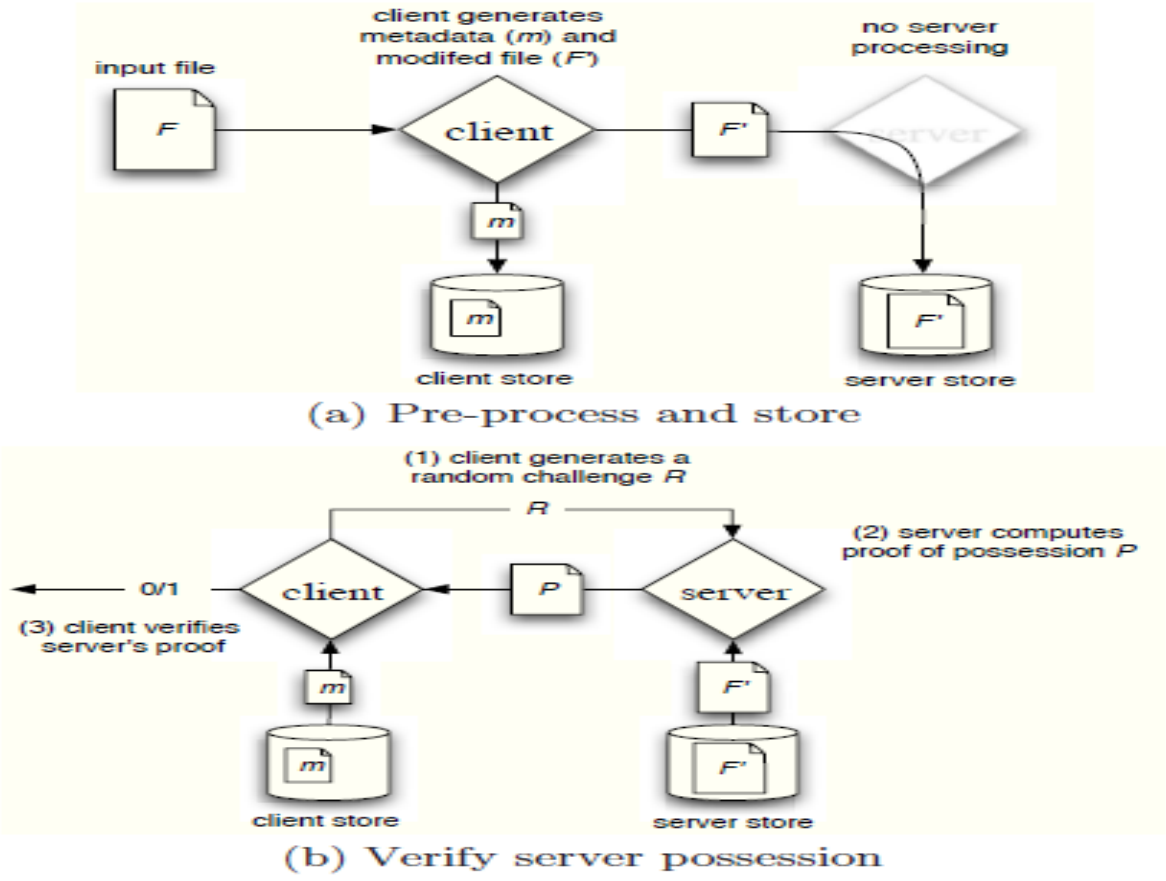


圖 六、PDP 完整性檢測系統運作示意圖[2]

4.1 函式定義

- 金鑰產生函式：由資料擁有者執行，讀入安全參數 k，生成金鑰對(pk, sk).

$$\mathrm{KeyGen}(1^k) \rightarrow (\mathrm{pk}, \mathrm{sk}).$$

- 完整性標籤計算函式：由資料擁有者執行，讀入金鑰對(pk, sk)和資料區塊 m，輸出此區塊的完整性標籤T_m.

$$\mathrm{TagBlock}(\mathrm{pk}, \mathrm{sk}, \mathrm{m}) \rightarrow (T_{i,m}, W_i).$$

- 證明碼計算函式：由雲端儲存裝置執行，讀入公開金鑰pk、資料區塊的集合 F、挑戰chal和資料區塊對應的完整性標籤集合Σ. 回傳證明ν.

$$\mathrm{GenProof}(\mathrm{pk}, \mathrm{F}, \mathrm{chal}, \Sigma) \rightarrow \nu.$$

- 驗證證明碼函式：由資料擁有者執行，讀入金鑰對(pk, sk)、chal、ν，回傳驗證成功與否。

$$\mathrm{CheckProof}(\mathrm{pk}, \mathrm{sk}, \mathrm{chal}, \nu) \rightarrow \{\mathrm{True}, \mathrm{False}\}.$$

4.2 系統概述

PDP 完整性檢測分成兩個階段：

- **Setup：**

資料擁有者在這個階段會利用 KeyGen()產生完整性檢測所需的金鑰對(key pair)。並且執行 TagBlock()對要上傳的檔案計算完整性標籤(integrity tag)。資料擁有者將金鑰對自己保存，並將資料檔案和完整性標籤上傳至雲端儲存裝置。

- **Challenge：**

資料擁有者希望驗證資料完整性時，隨機挑選要驗證的資料區塊、亂數產生秘密參數生成挑戰(chal, challenge)，傳給雲端儲存裝置。雲端儲存裝置收到挑戰，根據挑戰內容、儲存的資料檔案和對應的完整性標籤執行 GenProof()，產生證明碼(proof)，並回傳給資料擁有者。資料擁有者收到回傳的證明碼，利用 CheckProof()驗證，便可得知雲端儲存裝置是否擁有完整資料檔案。

4.3 建置 S-PDP (strong-PDP)

以下詳細介紹 S-PDP 兩個階段各函式的運作情形。

- **Preliminaries：**

$$\begin{cases} p = 2p' + 1 \\ q = 2q' + 1 \end{cases}$$ 都是安全的大質數，$N = pq$.

g: generator of QR_N(Quadratic Residue, cyclic subgroup of Z_N^* of order $p'q'$).

k, l, λ 是安全參數，其中 λ 是正整數。

pseudorandom function： $f : \{0,1\}^k \times \{0,1\}^{\log_2(n)} \to \{0,1\}^l$.

pseudorandom permutation： $\pi : \{0,1\}^k \times \{0,1\}^{\log_2(n)} \to \{0,1\}^{\log_2(n)}$.

H 為 cryptographic hash function.

- **Setup：**
  - ✧ **KeyGen**$(1^k) \to (pk, sk)$：

    產生金鑰對(pk, sk) $\begin{cases} pk = (N, g) \\ sk = (e, d, v)' \end{cases}$

    其中$\{0,1\}^k \to^R v$,

    且$ed \equiv 1 (\bmod p'q')$, $e > \lambda$ and $d > \lambda$.

  - ✧ **TagBlock**$(pk, sk, m, i) \to (T_{i,m}, W_i)$：

    資料擁有者利用公鑰$pk = (N, g)$、私鑰$sk = (d, v)$、切成n個區塊的資料m_i及區塊索引i為輸入值來產生完整性標籤。

    先計算$W_i = v||i$.

    再計算完整性標籤$T_{i,m_i} = (h(W_i) \cdot g^{m_i})^d mod\ N$.

    輸出(T_{i,m_i}, W_i).

- **Challenge：**

資料擁有者要求雲端儲存裝置證明資料的完整性時，會先發出挑戰$(\text{chal} = (\text{c}, \text{k}_1, \text{k}_2, g_\text{s}))$，

c為要檢查的區塊個數$1 \le \text{c} \le \text{n}$.

$\begin{cases} i_j = \Pi_{k_1}(j), \text{為要挑選的 c 個 blocks 的 indices} \\ a_j = f_{k_2}(j), \text{為計算 proof 實各 block 的參數} \end{cases}$，其中

k_1 與$\text{k}_2 =_\text{R} \{0,1\}^\text{k}$是隨機取得長度為 k 的二元字串。

$g_\text{s} = g^s$,

$\text{s} \leftarrow^\text{R} Z_N^*$為資料擁有者自己保存的挑戰私密資訊。

✧ **GenProof**$(\mathbf{pk}, \mathbf{F}, \mathbf{chal}, \mathbf{\Sigma}) \rightarrow \boldsymbol{\nu}$：

雲端儲存裝置收到挑戰後，根據挑戰的資訊、儲存的資料檔案以及完整性標籤計算證明$\nu = (\text{T}, \rho)$,

$$\text{T} = \text{T}_{\text{i}_1, m_{i_1}}^{a_1} \cdot \ldots \cdot \text{T}_{\text{i}_c, m_{i_c}}^{a_c} = \left(h(W_{i_1})^{a_1} \cdot \ldots \cdot h(W_{i_c})^{a_c} \cdot g^{a_1 m_{i_1} + \cdots + a_c m_{i_c}}\right)^d \bmod N.$$

$$\rho = \text{H}\left(g_\text{s}^{a_1 m_{i_1} + \cdots + a_c m_{i_c}} \bmod N\right).$$

✧ **CheckProof**$(\mathbf{pk}, \mathbf{sk}, \mathbf{chal}, \boldsymbol{\nu}) \rightarrow \{\boldsymbol{true}, \boldsymbol{false}\}$：

資料擁有者收到雲端儲存裝置回傳的證明碼、自己儲存的私密資訊以及提出挑戰的私密資訊，驗證證明碼，並確認資料的完整性。

$$\begin{cases} \text{pk} = (\text{N}, g) \\ \text{sk} = (\text{e}, v) \\ \text{chal} = (\text{c}, \text{k}_1, \text{k}_2, \text{s}) \end{cases},$$

$\nu = (\text{T}, \rho)$,

計算$\tau = \dfrac{T^e}{h(W_{i_j})^{a_j}} \bmod N = g^{a_1 m_{i_1} + \cdots + a_c m_{i_c}} \bmod N$

檢驗$\text{H}(\tau^\text{s} \bmod \text{N}) = \rho$是否成立，若成立則代表驗證通過回傳 true，否則回傳 false。

4. 4 E-PDP (efficient-PDP)

為原 PDP (S-PDP)的變形，藉由改變 S-PDP 中的參數，簡化使用者端和伺服器端的運算。但同時也犧牲了部份的安全性，伺服器在一定程度上可以偽造證明碼以通過使用者端的驗證。

系統建置和 S-PDP 幾乎相同，除了

$$\text{PRF } f_{k_2}(j) \rightarrow a_j, 1 \le j \le c.$$

修改為

$$a_j = 1,\ 1 \le j \le c.$$

則受到影響的兩個函式：

- **GenProof**$(\mathbf{pk}, \mathbf{F}, \mathbf{chal}, \mathbf{\Sigma}) \rightarrow \boldsymbol{\nu}$：

  $$T = T_{i_1, m_{i_1}}^1 \cdot \ldots \cdot T_{i_c, m_{i_c}}^1 = \left(h(W_{i_1})^1 \cdot \ldots \cdot h(W_{i_c})^1 \cdot g^{m_{i_1}+\cdots+m_{i_c}}\right)^d \bmod N,$$

  $$\rho = H\left(g_s^{m_{i_1}+\cdots+m_{i_c}} \bmod N\right).$$

- **CheckProof**$(\mathbf{pk}, \mathbf{sk}, \mathbf{chal}, \boldsymbol{\nu}) \rightarrow \{\boldsymbol{true}, \boldsymbol{false}\}$：

  計算$\tau = \frac{\tau^e}{h(W_{i_j})^1} \bmod N = g^{m_{i_1}+\cdots+m_{i_c}} \bmod N.$

E-PDP 可以將使用者端和伺服器端的運算簡化到僅一個指數運算。但是，E-PDP 只能夠保證伺服器擁有檔案區塊組合，而不是每一個區塊。若惡意伺服器將檔案區塊(file block)的所有組合預先儲存，則有可能偽造證明碼(proof)，通過使用者的驗證。不過，只要使用者使用較大的檔案區塊數目(n)，並且每次挑戰要求的區塊都不相同，則伺服器必須預存$\binom{n}{c}$種的區塊組合，是為一龐大數目。例如：$n = 1000, c = 101, \binom{n}{c} \approx 10^{140}$.

4.5 公開驗證(public verification)

另外，S-PDP 也有公開驗證的變形：不僅資料擁有者，所有的使用者都可以對資料進行完整性檢測。相較於 S-PDP 這個變形會增加網路傳輸的成本。以下為此變形系統的建置過程，我們以 S-PDP 的建置過程為範本，將修改的地方以**粗體**標示：

- Setup：
  - ✧ KeyGen$\left(1^{k}\right) \rightarrow (pk, sk)$：

    產生金鑰對$(\mathrm{pk}, \mathrm{sk})$ $\begin{cases} \mathrm{pk} = (\mathrm{N}, \mathrm{g}, \mathbf{e}) \\ \mathrm{sk} = (\mathrm{d}) \end{cases}$，

    且$\mathrm{ed} \equiv 1(\bmod \mathrm{p'q'}),\ \mathrm{e} > \lambda \text{ and } \mathrm{d} > \lambda.$

  - ✧ TagBlock$(\mathrm{pk}, \mathrm{sk}, m, i) \rightarrow \left(\mathrm{T}_{i,m}, \mathrm{W}_i\right)$：

    資料擁有者利用公鑰$\mathbf{pk} = (\mathbf{N}, \mathbf{g})$、私鑰$\mathbf{sk} = (\mathbf{d})$、切成n個區塊的資料$m_i$及區塊索引$i$為輸入值來產生完整性標籤，**其中**$\boldsymbol{m_i < \lambda/2, \forall i}$。

    先計算$\mathbf{W_i} = \boldsymbol{w_v(i)}$，**其中**，$\begin{cases} \boldsymbol{PRF}\ \boldsymbol{w}: \{\mathbf{0},\mathbf{1}\}^{\mathbf{k}} \times \{\mathbf{0},\mathbf{1}\}^{\mathbf{log_2(n)}} \rightarrow \{\mathbf{0},\mathbf{1}\}^{\boldsymbol{l}} \\ \{\mathbf{0},\mathbf{1}\}^{\mathbf{k}} \rightarrow^{\boldsymbol{R}} \boldsymbol{v} \end{cases}$.

    再計算完整性標籤$\mathrm{T}_{i,m_i} = (h(\mathrm{W}_i) \cdot g^{m_i})^{d} mod\ N.$

    輸出$\left(\mathrm{T}_{i,m_i}, \mathrm{W}_i\right)$.

    公開$\boldsymbol{v}$.

➢ Challenge：

資料擁有者要求雲端儲存裝置證明資料的完整性時，會先發出挑戰$(\text{chal} = (\text{c}, \text{k}_1, \text{k}_2))$，

c為要檢查的區塊個數$1 \le \text{c} \le \text{n}$.

$\begin{cases} i_j = \Pi_{k_1}(j), \text{為要挑選的 c 個 blocks 的 indices} \\ a_j = f_{k_2}(j), \text{為計算 proof 實各 block 的參數} \end{cases}$，其中

k_1 與$\text{k}_2 =_{\text{R}} \{0,1\}^{\text{k}}$是隨機取得長度為 k 的二元字串。

✧ GenProof$(\text{pk}, \text{F}, \text{chal}, \Sigma) \to \nu$：

雲端儲存裝置收到挑戰後，根據挑戰的資訊、儲存的資料檔案以及完整性標籤計算證明$\nu = (\text{T}, \boldsymbol{M})$,

$$\text{T} = \text{T}_{\text{i}_1, m_{i_1}}^{a_1} \cdot \ldots \cdot \text{T}_{\text{i}_c, m_{i_c}}^{a_c} = \left(h\left(W_{i_1}\right)^{a_1} \cdot \ldots \cdot h\left(W_{i_c}\right)^{a_c} \cdot g^{a_1 m_{i_1} + \cdots + a_c m_{i_c}}\right)^{d} \bmod N.$$

$$\boldsymbol{M = a_1 m_{i_1} + \cdots + a_c m_{i_c}.} \quad \textbf{(2)}$$

✧ CheckProof$(\text{pk}, \text{sk}, \text{chal}, \nu) \to \{true, false\}$：

資料擁有者收到雲端儲存裝置回傳的證明碼、自己儲存的私密資訊以及提出挑戰的私密資訊，驗證證明碼，並確認資料的完整性。

$\begin{cases} \textbf{pk} = (\text{N}, \boldsymbol{g}, \boldsymbol{e}) \\ \textbf{chal} = (\text{c}, \mathbf{k_1}, \mathbf{k_2})' \end{cases}$

$\nu = (\text{T}, \boldsymbol{M})$,

計算$\tau = \dfrac{T^e}{h\left(W_{i_j}\right)^{a_j}} \bmod N = g^{a_1 m_{i_1} + \cdots + a_c m_{i_c}} \bmod N.$

檢驗$((\boldsymbol{\tau = g^M}) \textbf{且} (|\boldsymbol{M}| < \boldsymbol{\lambda/2}))$是否成立，若成立則代表驗證通過回傳 true，否則回傳 false。

但這個變形的雲端儲存裝置不能夠知道$\text{p}'\text{q}' = order\ of\ (\text{QR}_{\text{N}})$. 否則可以利用 pk 中的 e 和$\text{p}'\text{q}'$計算得到sk 的 d，進而偽造完整性標籤。

這樣會使得上式(2)中的M不能先mod p'q'，將會大幅增加傳輸證明碼v時的成本。

第 5 章 系統設計與實作

5.1 系統概述

我們的系統-SDEMR (Secure Distributed EMR)，顧名思義，針對電子醫療記錄所設計的安全資料庫系統。在設置階段，管理機構(Authority)會配發金鑰資訊給所有的使用者(病患、醫事人員)，和代理伺服器(Proxy)。由於醫療資料實屬病患的隱私，但也需要醫學專業的背景來撰寫；是故，我們的情境假設新增和修改記錄的操作必須有病患及醫事人員在場，如：醫師。另外，病患可以獨自查詢自己的醫療記錄，醫師也可以查詢相關診療記錄以利醫學、病例研究。我們的系統的操作概念如下圖七所示：

1） 管理機構配發相關金鑰資訊給病患、醫師和代理伺服器。

2） 病患在醫師的陪同下上傳醫療病歷記錄。

3） 病患或是醫師可以查詢相關的醫療記錄

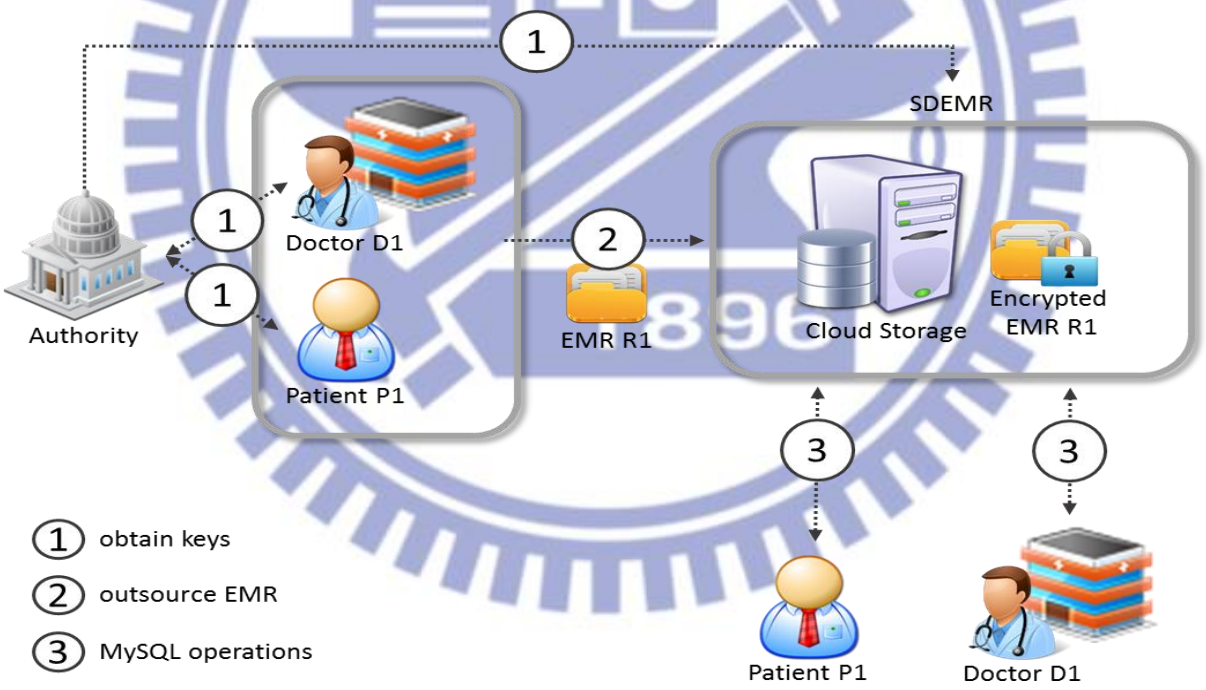


圖 七、系統操作概念

我們的系統在新增和查詢醫療資料時，會自動地進行完整性相關資訊的計算、驗證資料的完整性；以確保使用者查詢的資料沒有遭到竄改。

5.2 系統架構與重要元件

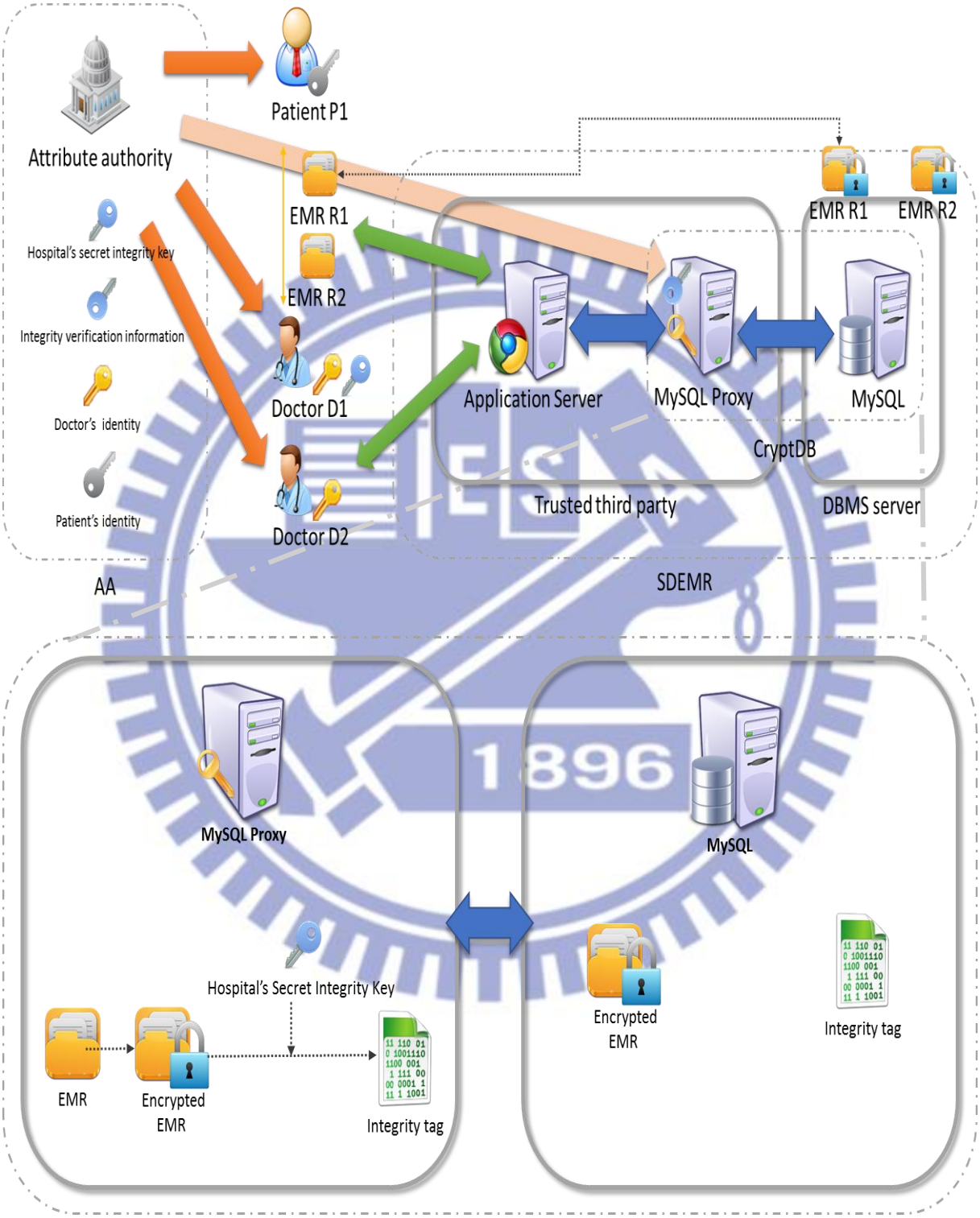


圖 八、系統架構與運作概述

如上圖八，我們的系統主要區分成三個部份：瀏覽器伺服器(Application Server)、代理伺服器(MySQL Proxy)和後端儲存裝置(DBMS Server)；致力於防備惡意的外部攻擊者和好奇的後端儲存裝置提供廠商：降低系統操作時的資訊洩漏，並且能夠偵測資料遭到惡意竄改。

- **瀏覽器伺服器**

  是我們的系統和使用者之間的互動介面，負責處理使用者的連線、登入、資料的呈現等相關任務。以安全套接層協議(SSL, Secure Sockets Layer)，保護傳輸資訊的私密性。

- **代理伺服器**

  負責加解密金鑰的管理，資料庫操作指令的判讀、改寫、加解密常數以及記錄後端儲存裝置的密文表單結構狀況，並且負責新增和查詢時，資料的完整性檢測相關運算。

- **後端儲存裝置**

  儲存代理伺服器加密後的醫療資料。利用 MySQL 的使用者自定義函式(UDF, User Define Function)，支援密文的同態運算、最外層(RND)密文的解密等動作。

和我們採用的元件—CryptDB 類似，我們的環境設定瀏覽器伺服器和代理伺服器是可以完全信任的第三方；後端儲存裝置則是可以被半信任的，意即它會遵照我們的要求執行指令，但也會在計算、傳輸的過程，企圖得到更多的資訊。

5.3 系統建置

5.3.1 系統設置階段

系統設置階段，管理機構(Authority)如：衛生署會分發給每位病患健保 IC 卡，包含病患驗證個人身份的私密金鑰；另外發給醫療機關計算完整性標籤用的私密金鑰。醫療機關也分發給每位醫事人員如：醫師、護士、藥劑師…等醫事人員 IC 卡，包含驗證個人身份的私密金鑰和代表醫療機關計算完整性標籤使用的私密金鑰(Hospital's secret integrity key)。同時，將驗證完整性所需的金鑰資訊(Integrity verification information)傳送給代理伺服器，並確認代理伺服器有收到並且儲存。

5.3.2 使用情境

我們利用 IC 卡中的金鑰來確認使用者的身份、在場。使用者(病患或醫事人員)將 IC 卡插入讀卡機，以行動裝置或是個人電腦透過安全連線，連線到本系統；瀏覽器伺服器會提供網頁介面，引導使用者進行操作。使用者操作網頁介面，瀏覽器伺服器再傳輸指令至代理伺服器操作醫療資料庫。

由於醫療資料需要長時間留存備查的特性，本系統不支援刪除醫療資料的操作，修改的操作僅限於基本資料如：病患基本資料表單、緊急聯絡人表單的異動。其他的表單，一律支援新增、排序、整數相加、比對查詢資料的操作。以下，列舉三個最常見的情境概略說明系統的運作。首先，網頁會引導使用者登入，並且取得使用者 IC 卡中的私密金鑰，藉以確認使用者的身分。

情境一、病患和醫師看診結束，醫師新增醫療記錄：

醫師透過本系統網頁撰寫醫療病歷記錄，由病患確認後送出。此筆醫療記錄和新增記錄的 MySQL 指令，自瀏覽器伺服器傳輸至代理伺服器。代理伺服器改寫指令，並利用加密金鑰對醫療記錄做洋蔥加密。再以醫事人員 IC 卡中：醫療機關為單位的完整性檢測私密金鑰，對密文計算完整性標籤。待所有動作完成後，將加密後的醫療資料、完整性標籤一併傳輸至後端儲存裝置。

情境二、使用者（病患或醫事人員）希望查詢相關醫療記錄：

使用者透過網頁選擇想要比對查詢（如：用病患名字做篩選）的醫療資料。代理伺服器接收到查詢請求。指示後端儲存裝置利用使用者自定義函式（UDF）剝去洋蔥加密法的 RND 層，進行密文比對的操作，自後端儲存裝置傳回符合比對條件的資料至代理伺服器。代理伺服器收到回傳結果，先對剝去 RND 層的密文做完整

性檢測，通過檢測才解密並回傳得到的醫療資料到網頁前端。

情境三、病患基本資料異動，希望醫事人員協助修改基本資料：

醫事人員經由網頁查詢得病患的基本資料，利用網頁的修改功能，進行資料修改。在病患確認無誤之後，傳輸至代理伺服器。代理伺服器收到資料修改的請求，會以刪除舊資料、新增（系統運作就如情境一的情形）修改好的資料來完成修改的操作。

5.3.3 完整性檢測機制

我們系統的完整性檢測機制的主要概念在於：將新增的醫療病歷資料加密後，對密文計算完整性標籤(tag)，利用儲存的密文資料和標籤便可以計算證明碼(proof)以檢測資料有無被更動。

CryptDB 支援兩種資料形態：int 和 text；在洋蔥加密法的設計中，Onion Add 和 Onion Search 分別為 int 和 text 資料型態個別獨有。Onion Eq 和 Onion Ord 則是 int 和 text 都有的洋蔥密文形式。於是，我們將自這兩種洋蔥密文挑選計算完整性標籤的目標。

我們的系統中，最常使用”條件查詢資料”(SELECT WHERE)的操作。CryptDB 在使用條件查詢時，依照條件的形式來選擇需要使用的洋蔥加密密文(Onion Eq 和 Onion Ord)。如下式(3)為”比對相等查詢”，CryptDB 採用 ***fieldB*** 的 Onion Eq 來比對值是否和 ***1*** 的密文相等。另一個常見的是”比較大小查詢”如下式(4)，CryptDB 會採用 ***fieldB*** 的 Onion Ord 的值和 ***1*** 的密文比較大小；取得符合大小條件的列(row)，但是仍然自這些列中選取 ***fieldA*** 的 Onion Eq 密文來解密，得到明文資料。Onion Eq 在我們的系統中，是解密呈現給使用者的主要密文來源；因此，我們統一對 Onion Eq 密文做完整性檢測。

$$\textbf{SELECT } \boldsymbol{fieldA} \textbf{ FROM } \boldsymbol{table} \textbf{ WHERE } \boldsymbol{fieldB} = \mathbf{1}; \qquad \textbf{(3)}$$

$$\textbf{SELECT } \boldsymbol{fieldA} \textbf{ FROM } \boldsymbol{table} \textbf{ WHERE } \boldsymbol{fieldB} > \mathbf{1}; \qquad \textbf{(4)}$$

選取 Onion Eq 密文作為我們實作完整性檢測的目標之後，我們還要考慮選取 RND、DET、JOIN、PLAIN VALUE 中的哪一層來計算完整性標籤。考慮 CryptDB 使用洋蔥加密法，如果我們直接對最內層的明文(PLAIN VALUE)計算完整性標籤，則每次計算證明時，必須先解密得到明文，發現完整性檢驗未通過時，已耗費資源解密。另外，RND 層的密文無法執行任何資料庫操作，往往送到代理伺服器時已經被剝除，例如當我們執行上式(3)之後，代理伺服器會要求後端儲存裝置將 ***table.fieldB*** 整行(column)的 RND 層剝除，以利進行和 ***1*** 的密文比對的操作。因此，雖然 Onion Eq 和 Onion Ord 的最外層還包有安全性最高的 RND 層，但是我們不使用這一層的密文來計算完整性標籤。至於從 DET

或是 JOIN 層著手，我們認為沒有太大的差別，但是系統中鮮少關於連結(JOIN)的操作指令，是故，我們最後選擇對 DET 層的密文計算完整性標籤。

- **新增資料：**

  如下圖九示意：代理伺服器收到醫生和病患新增的醫療病歷資料之後，先根據相關表單(table)、行(column)、密文形式(onion)及加密層(layer)以(1)式計算加密金鑰，接著將資料以計算出來的金鑰層層加密；其中，加密至 DET 層時，會先利用醫療機關計算完整性標籤的私密金鑰計算 DET 層密文的完整性標籤，才繼續將 DET 層再包上 RND 層。最後將 RND 的密文和完整性標籤一起送到後端儲存裝置儲存。

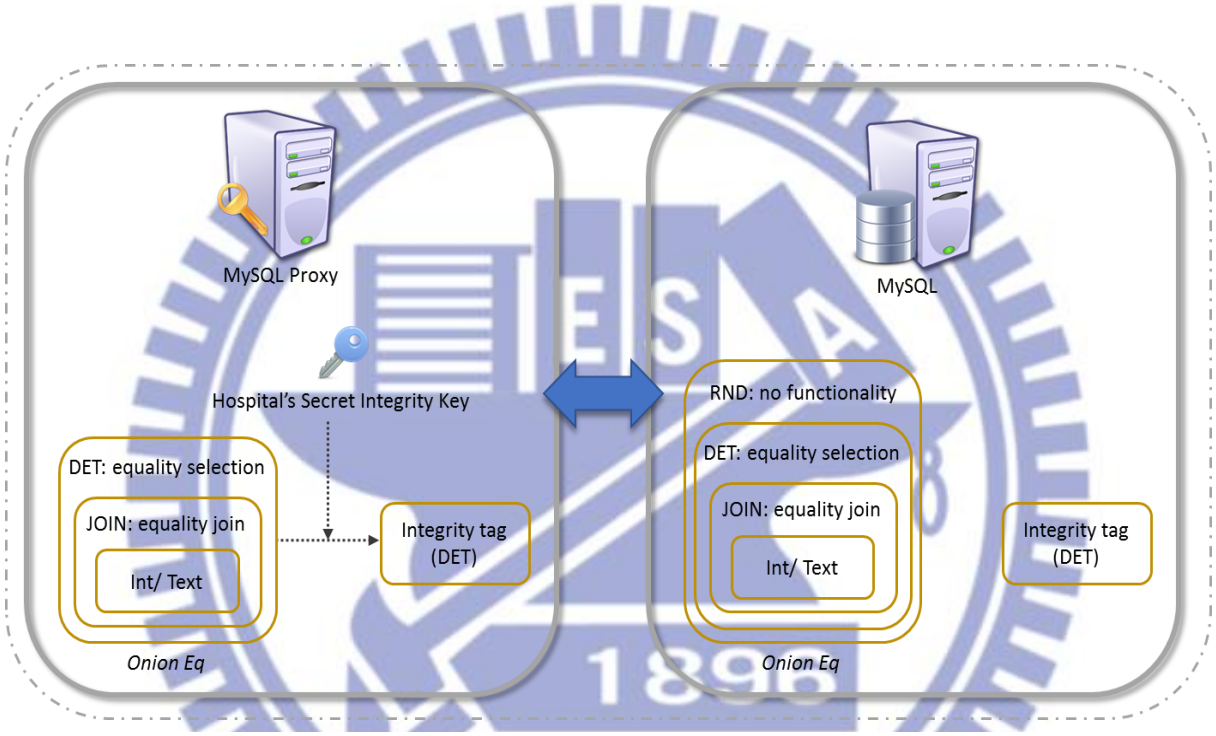


圖 九、完整性檢測-新增資料運作示意圖

➢ **查詢資料：**

圖十，代理伺服器收到查詢資料的操作指令，先判斷需不需要剝除密文資料的 RND 層：若指令中有比對相等的條件，代理伺服器會先要求後端儲存裝置利用使用者自定義函式(UDF)對密文資料進行解密，剝除 RND 層，得到剩下 DET 層的密文資料。代理伺服器再改寫比對值成 DET 的密文，以要求後端儲存裝置回傳符合條件的密文資料和對應的完整性標籤。另外一種情況，若指令只是一般的查詢操作，則後端儲存裝置將選擇的欄位(field)的密文資料和對應的完整性標籤傳回，由代理伺服器先對密文資料剝除 RND 層，再對 DET 層密文資料進行完整性檢測。

我們的系統假設代理伺服器可以完全被信任，交由代理伺服器完成 PDP 驗證完整性的過程：隨機生成挑戰資訊，對密文資料產生證明碼，然後驗證證明碼是否正確，以確定後端儲存裝置回傳的密文和標籤無誤，最後才將密文資料解密。

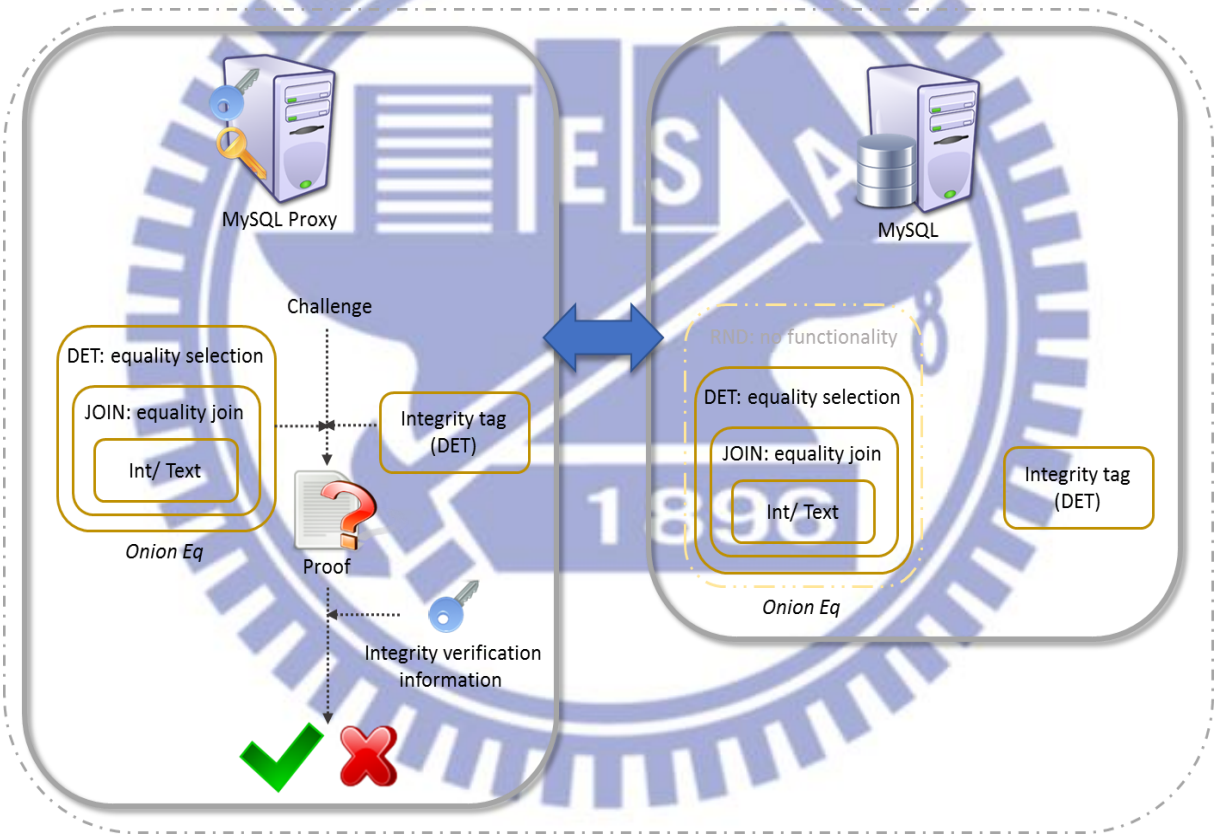


圖 十、完整性檢測-查詢資料運作示意圖

➢ **修改資料：**

我們的系統利用刪除舊有資料、再新增修改後資料的方式來完成修改資料的操作。不採用支援動態完整性檢測系統的原因有二：

1） 因應醫療病歷資料長時間備存的特性，修改資料的資料庫操作比較少見。在我們的系統中只支援病人基本資料、緊急聯絡人兩張表單執行修改的操作。

2） 維護支援動態完整性檢測的系統，動態計算完整性檢測的成本較高。另外，我們以欄位(field)為單位進行完整性檢測，每筆檢測資料的不大，直接重新計算新的完整性標籤成本也不會過高。

➢ **技術摘要：**

✧ **以欄位(field)為單位計算完整性標籤：**

如下圖十一示意：我們的系統以欄位(field)為單位實做完整性檢測。我們對每一個欄位的值計算完整性標籤，並置放於其後的欄位。相較以列(row)為單位計算進行完整性標籤，這樣的方法會增加許多的儲存完整性標籤的空間(正比於欄位數)。但是可以因應 CryptDB 的特性，調整層(layer)的加解密運算以行(column)為主；在洋蔥加密法中，內層（如 DET 層）的加密法，相同行的欄位以相同的金鑰加密，以利密文的資料庫操作。

test1

Id (INT)	Name (TEXT)
1	Alice

table_GPMIFXLMEQ

oEq	oOrd	oAdd	salt	IC_tag	oEq	oOrd	oSWP	salt	IC_tag
x2b82	xcb85	xdcf5	x27c2	x7952	x8123	xdle7	x29b0	x7db5	xa248

圖 十一、test1 表單和系統中密文表單對應情形(以欄位為單位計算完整性標籤)

若是我們以列為計算完整性標籤的單位，首先必須考慮：整列的密文資料以哪一層作為計算完整性標籤的目標。假設我們以最直覺的方法，用最高層(Onion Eq: RND, Onion Ord: RND, Onion Add: HOM, Onion Search: SEARCH)來計算。當系統收到條件查詢的操作指令如上式(3)，必須剝除 ***fieldB*** 的 Onion Eq 一整行的密文至 DET 層，將會造成整列的資料因為少數欄位密文層的不同，而和完整性標籤不一致。或許必須重新計算所有受到變動的資料的完整性標籤；

抑或是在每一次的完整性檢測過程，都必須先將列中的每一個欄位資料調整至計算完整性標籤時的最高加密層，造成許多額外的加解密運算成本。於是，我們以欄位為計算完整性標籤的單位。

另外，對於查詢部分欄位時，我們也只需要挑選被查詢的欄位進行完整性檢測，而不是整列的資料。由於完整性檢測牽涉的資料量小，同時也默許我們在每次的完整性檢測對挑選的整個欄位資料進行檢測，而不需要挑選資料區塊，消彌檢測疏漏的機會，確保醫療資料的可信度。

在上述修改情境的資料庫操作中，也因為我們單位完整性檢測牽涉的資料量小，因此可以直接利用刪除和新增，取代在醫療資料庫中少見的修改操作。而不需要另外維持支援動態完整性檢測的系統。

✧ **完整性檢測金鑰管理**

PDP 完整性檢測的金鑰為一組金鑰對，分別是負責計算完整性標籤以及驗證證明碼的金鑰。前者必須保持私密，否則擁有金鑰便可以偽造完整性標籤。後者則視系統應用情形，可以是私密的，意即擁有金鑰者才能夠驗證；或是公開的，所有人都可以驗證。不論是哪一種情形，計算完整性標籤的私密金鑰，必定有對應的驗證證明碼的金鑰。

我們的系統可以採取以下的方式管理完整性檢測的金鑰對：

- 整個系統採用一對完整性檢測金鑰，置於代理伺服器：由代理伺服器在新增醫療資料時，自動計算完整性標籤；查詢醫療資料時，自動驗證完整性。
- 系統採用多對完整性檢測金鑰，依照使用者／醫療機關配發：
  新增醫療資料時，使用者／醫療機關將計算完整性標籤的私密金鑰傳送給代理伺服器計算完整性標籤；並將對應的驗證完整性證明碼金鑰置於代理伺服器，以供查詢資料時驗證完整性。
- PDP 完整性檢測變形-公開驗證：
  類似上述系統採用多對完整性檢測金鑰的做法，將計算完整性標籤的私密金鑰交由使用者／醫療機關保管；使用者在查詢資料的時候，代理伺服器會利用公開的驗證資訊來檢測資料完整性。但是這個方法用來計算完整性標籤的檔案區塊大小有比較多的限制。

我們的系統以醫療機關為單位來管理完整性檢測使用的金鑰：將計算完整性標籤部份的私密金鑰置於醫事人員 IC 卡中。希望達到新增醫療資料時，必須確認醫事人員在場的情境假設。

另外，醫師和病患可以獨自查詢資料。然而，病患查詢資料的時候，不會擁有醫療機關的驗證完整性證明碼的金鑰；但是我們仍然希望資料在呈現給任何使用者之前，可以先檢測完整性。是故，我們的系統將醫療機關對應的完整性驗證金鑰置於代理伺服器，不論醫事人員或是病患在查詢資料的時候，代理

伺服器會代為進行資料的完整性驗證。

相較於以醫事人員為單位，這樣的金鑰分配可以降低代理伺服器管理驗證完整性證明碼金鑰的數量、減輕負擔。

✧ **代理伺服器進行完整性檢測運算：**

我們系統的完整性檢測機制，交由代理伺服器來發出挑戰、計算證明碼和驗證證明碼。需要自後端儲存裝置取回完整的資料和相關的完整性標籤至代理伺服器。乍看之下，這樣的運作模式已經喪失 PDP 完整性檢測的優點：不需要回傳整份的資料檔案以降低傳輸成本。但是，由於代理伺服器必須負責資料最後階段的解密動作，後端儲存裝置無論如何都必須將密文（原檔案資料）回傳至代理伺服器進行解密。是故，我們的系統將完整性證明碼的計算(GenProof)自後端儲存裝置轉移到代理伺服器；除了完整性標籤，並沒有額外增加其他傳輸成本。

5.4 實作結果

5.4.1 醫療資料參考來源

我們參考行政院衛生福利部的電子病歷管理系統，依據”電子病歷標準管理系統”中的”單張基本格式”常見表單，整理新建八張表單、創建五十位病患並且隨機生成對應的醫療病歷資料來模擬我們的醫療資料庫，以供驗證上述的功能。

為了醫療資料擬真，表單之間我們考慮其中醫療記錄的關係，如：病患和個人基本資料表單為一對一的關係，和處方籤表單則是一對多的關係。整理如下表三：

表一、SDEMR 資料統計

表單名稱	對應電子病歷 單張基本格式	資料對應使用者數量	總資料量
Patient_Basic_Data	病歷首頁單	Exact one	50
Emergency_Contact	病歷首頁單	Many (1~2)	50
Policlinic_Record	門診複診單	Many (2~5 as X)	250
Prescription	門診處方籤	Many (X)	250
SOAP_Record	病程記錄單	Many (X)	250
Resident_Note	入院記錄單	Many (as Y <= X)	67
Vital_Signs_Measurments	生命徵象測量記錄單	Many (5~10 times Y)	805
Faculty		Exact one	50

我們的資料以表單為主軸來建立，表單之間以病患編碼（Patient_ID）和醫事人員編碼（Faculty_ID）連接。每個病患在個人基本資料表單有對應的一筆資料，記錄病患基本資料及病患編碼。其他表單的資料都會利用這個編碼和病患做連結，如病患每次看診，會在病程記錄表單（SOAP_Record）記錄一筆診斷資料和此病患的編碼；在門診處方籤表單（Prescription）記錄一筆用藥資料，還有對應的病患編碼。當這個病患回診的時候，

醫師會利用病患編碼查詢資料庫中，相關的醫療記錄，以協助這次診斷。下圖十二為我們的模擬醫療資料庫中，各表單之間的相依關係：病患基本資料(Patient_Basic_Data)和醫事人員(Faculty)兩張表單分別記錄病患和醫事人員和其編號(Pid, Fid)的一對一對應關係。其餘的表單則以此編號和病患、醫事人員連結。

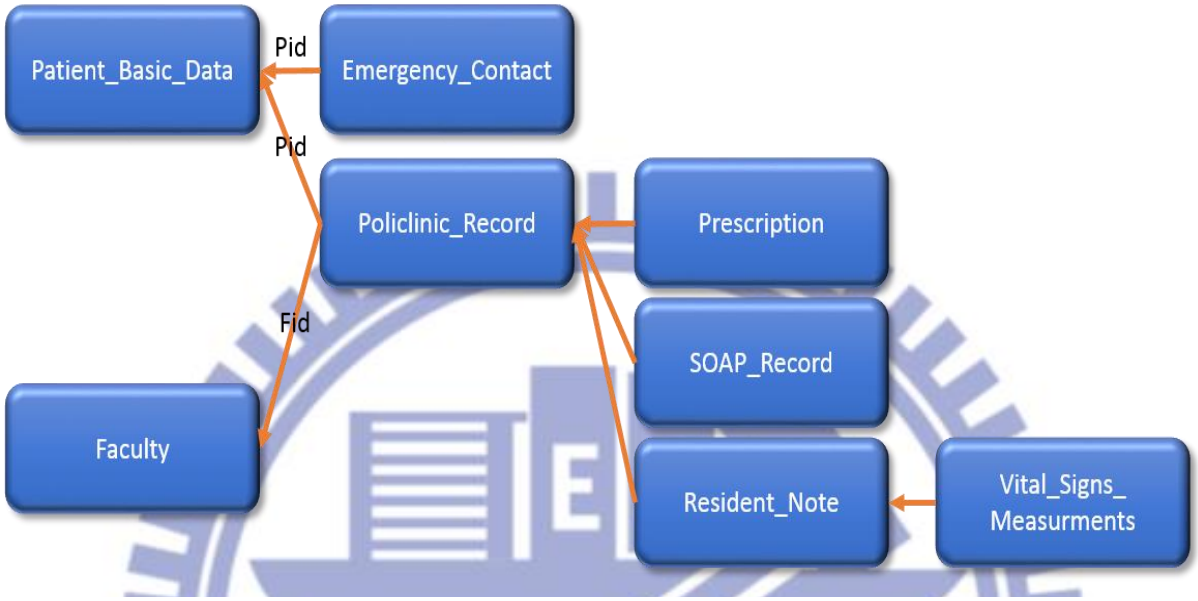


圖 十二、各表單之間的相依關係

5.4.2 實驗結果

我們以醫療資料庫中的處方籤表單(Prescription)為例，選取一個整數型態(INT)和三個文字型態(TEXT)的欄位，分別是：處方籤編號(Pr_Id, INT)、病患編號(Pid, TEXT)、病患姓名(Patient_Name, TEXT)和藥名(Drug_Name, TEXT)四個欄位的資料來驗證我們的完整性檢測機制。

- **新增資料：**
  - ✧ 圖十三：透過代理伺服器連入 MySQL，新增處方籤表單的資料。

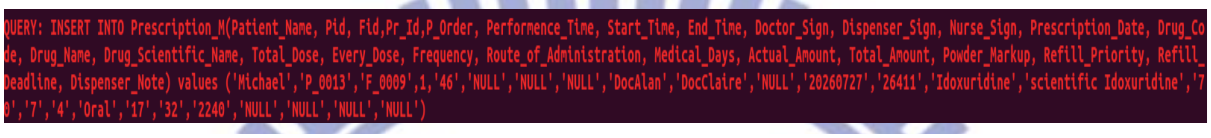


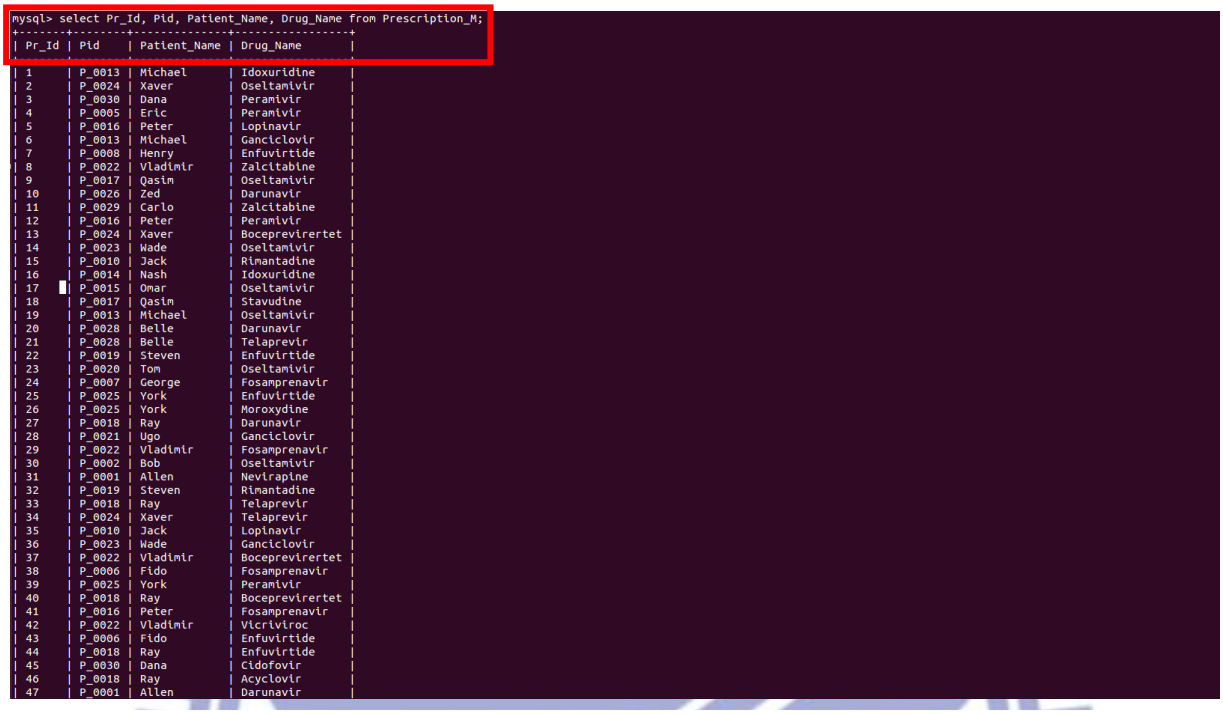
圖 十三、終端機介面-新增資料

- ✧ 圖十四：代理伺服器的訊息，改寫插入的操作指令：原操作指令中的每個欄位，除了 CryptDB 洋蔥加密法擴增的欄位以外，也另外新增了存放完整性標籤的欄位(IC_tag_XXX)；對應的插入值也有系統自動計算的完整性標籤值。

圖 十四、終端機介面-計算完整性標籤

➢ **查詢資料：**

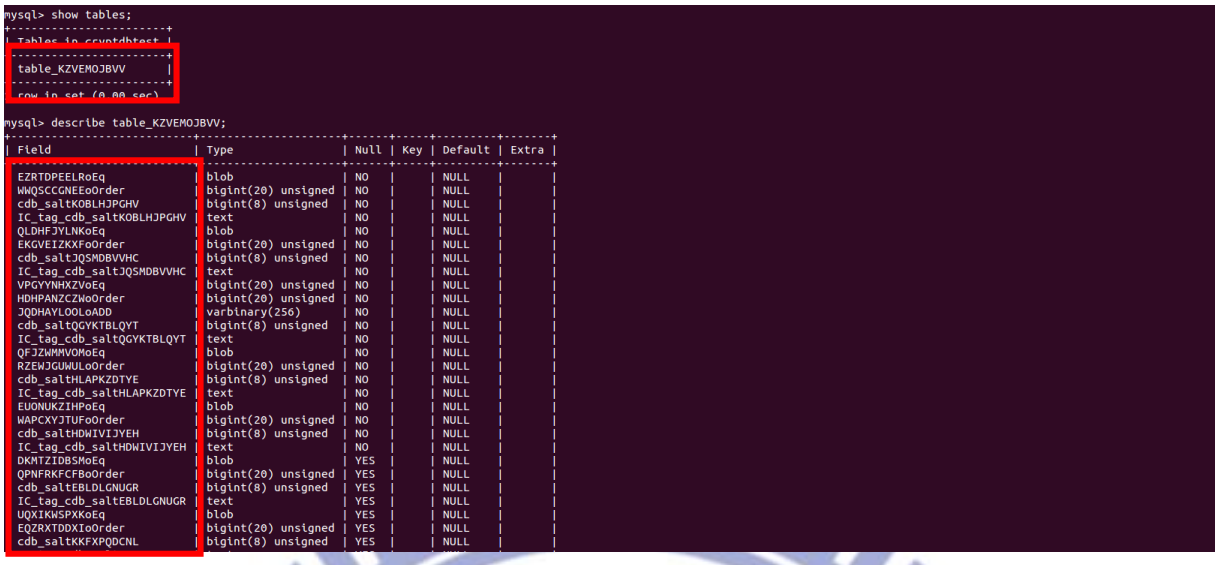
✧ 圖十五：透過代理伺服器連上 MySQL，查詢處方籤表單中的處方籤編號、病患編號、病患姓名和藥名（依序由左至右），可以成功解密得到資料回傳。



```
mysql> select Pr_Id, Pid, Patient_Name, Drug_Name from Prescription_M;
+-------+--------+--------------+-----------------+
| Pr_Id | Pid    | Patient_Name | Drug_Name       |
+-------+--------+--------------+-----------------+
| 1     | P_0013 | Michael      | Idoxuridine     |
| 2     | P_0024 | Xaver        | Oseltamivir     |
| 3     | P_0030 | Dana         | Peramivir       |
| 4     | P_0005 | Eric         | Peramivir       |
| 5     | P_0016 | Peter        | Lopinavir       |
| 6     | P_0013 | Michael      | Ganciclovir     |
| 7     | P_0008 | Henry        | Enfuvirtide     |
| 8     | P_0022 | Vladimir     | Zalcitabine     |
| 9     | P_0017 | Qasim        | Oseltamivir     |
| 10    | P_0026 | Zed          | Darunavir       |
| 11    | P_0029 | Carlo        | Zalcitabine     |
| 12    | P_0016 | Peter        | Peramivir       |
| 13    | P_0024 | Xaver        | Boceprevirertet |
| 14    | P_0023 | Wade         | Oseltamivir     |
| 15    | P_0010 | Jack         | Rimantadine     |
| 16    | P_0014 | Nash         | Idoxuridine     |
| 17    | P_0015 | Omar         | Oseltamivir     |
| 18    | P_0017 | Qasim        | Stavudine       |
| 19    | P_0013 | Michael      | Oseltamivir     |
| 20    | P_0028 | Belle        | Darunavir       |
| 21    | P_0028 | Belle        | Telaprevir      |
| 22    | P_0019 | Steven       | Enfuvirtide     |
| 23    | P_0020 | Tom          | Oseltamivir     |
| 24    | P_0007 | George       | Fosamprenavir   |
| 25    | P_0025 | York         | Enfuvirtide     |
| 26    | P_0025 | York         | Moroxydine      |
| 27    | P_0018 | Ray          | Darunavir       |
| 28    | P_0021 | Ugo          | Ganciclovir     |
| 29    | P_0022 | Vladimir     | Fosamprenavir   |
| 30    | P_0002 | Bob          | Oseltamivir     |
| 31    | P_0001 | Allen        | Nevirapine      |
| 32    | P_0019 | Steven       | Rimantadine     |
| 33    | P_0018 | Ray          | Telaprevir      |
| 34    | P_0024 | Xaver        | Telaprevir      |
| 35    | P_0010 | Jack         | Lopinavir       |
| 36    | P_0023 | Wade         | Ganciclovir     |
| 37    | P_0022 | Vladimir     | Boceprevirertet |
| 38    | P_0006 | Fido         | Fosamprenavir   |
| 39    | P_0025 | York         | Peramivir       |
| 40    | P_0018 | Ray          | Boceprevirertet |
| 41    | P_0016 | Peter        | Fosamprenavir   |
| 42    | P_0022 | Vladimir     | Vicriviroc      |
| 43    | P_0006 | Fido         | Enfuvirtide     |
| 44    | P_0018 | Ray          | Enfuvirtide     |
| 45    | P_0030 | Dana         | Cidofovir       |
| 46    | P_0018 | Ray          | Acyclovir       |
| 47    | P_0001 | Allen        | Darunavir       |
```

圖 十五、終端機介面–查詢資料，解密得明文

✧ 圖十六：不透過代理伺服器，直接連入 MySQL，可以看到處方籤表單和其欄位均以匿名呈現。

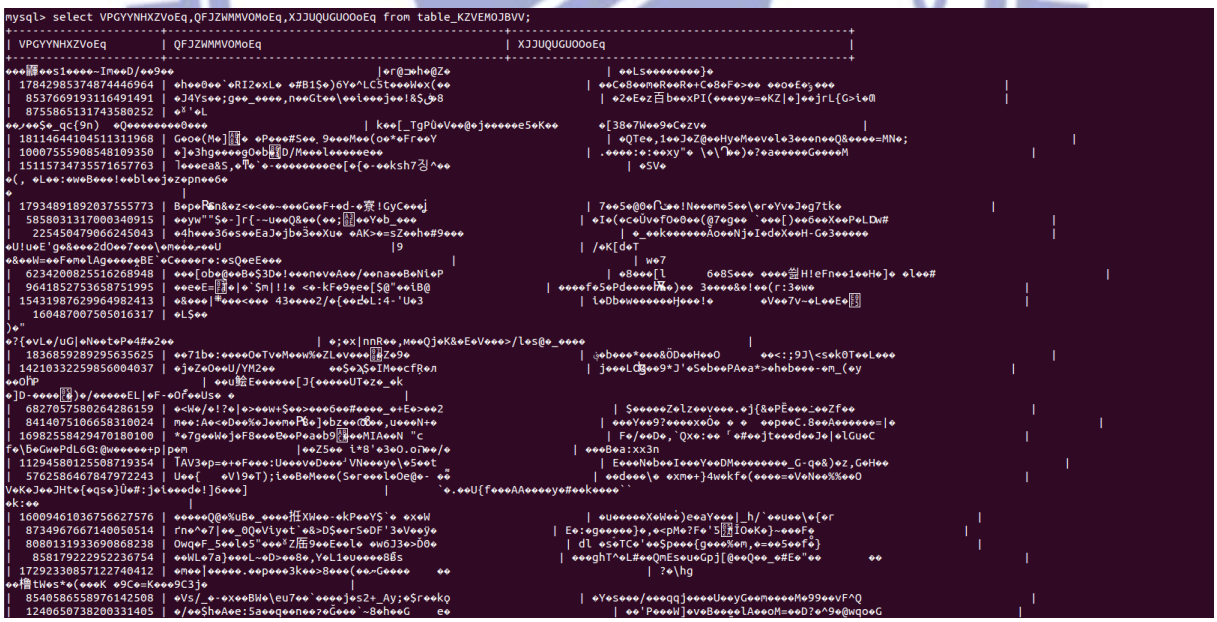


```
mysql> show tables;
+-----------------------+
| Tables_in_ccvntdbtest |
+-----------------------+
| table_KZVEMOJBVV      |
+-----------------------+
1 row in set (0.00 sec)

mysql> describe table_KZVEMOJBVV;
+---------------------------+---------------------+------+-----+---------+-------+
| Field                     | Type                | Null | Key | Default | Extra |
+---------------------------+---------------------+------+-----+---------+-------+
| EZRTDPEELRoEq             | blob                | NO   |     | NULL    |       |
| WWQSCCGNEEoOrder          | bigint(20) unsigned | NO   |     | NULL    |       |
| cdb_saltKOBLHJPGHV        | bigint(8) unsigned  | NO   |     | NULL    |       |
| IC_tag_cdb_saltKOBLHJPGHV | text                | NO   |     | NULL    |       |
| QLDHFJYLNKoEq             | blob                | NO   |     | NULL    |       |
| EKGVEIZKXFoOrder          | bigint(20) unsigned | NO   |     | NULL    |       |
| cdb_saltJQSMDBVVHC        | bigint(8) unsigned  | NO   |     | NULL    |       |
| IC_tag_cdb_saltJQSMDBVVHC | text                | NO   |     | NULL    |       |
| VPGYYNHXZVoEq             | bigint(20) unsigned | NO   |     | NULL    |       |
| HDHPANZCZWoOrder          | bigint(20) unsigned | NO   |     | NULL    |       |
| JQDHAYLOOLoADD            | varbinary(256)      | NO   |     | NULL    |       |
| cdb_saltQGYKTBLQYT        | bigint(8) unsigned  | NO   |     | NULL    |       |
| IC_tag_cdb_saltQGYKTBLQYT | text                | NO   |     | NULL    |       |
| QFJZWMMVOMoEq             | blob                | NO   |     | NULL    |       |
| RZEWJGUWULoOrder          | bigint(20) unsigned | NO   |     | NULL    |       |
| cdb_saltHLAPKZDTYE        | bigint(8) unsigned  | NO   |     | NULL    |       |
| IC_tag_cdb_saltHLAPKZDTYE | text                | NO   |     | NULL    |       |
| EUONUKZIHPoEq             | blob                | NO   |     | NULL    |       |
| WAPCXYJTUFoOrder          | bigint(20) unsigned | NO   |     | NULL    |       |
| cdb_saltHDWIVIJYEH        | bigint(8) unsigned  | NO   |     | NULL    |       |
| IC_tag_cdb_saltHDWIVIJYEH | text                | NO   |     | NULL    |       |
| DKMTZIDBSMoEq             | blob                | YES  |     | NULL    |       |
| QPNFRKFCFBoOrder          | bigint(20) unsigned | YES  |     | NULL    |       |
| cdb_saltEBLDLGNUGR        | bigint(8) unsigned  | YES  |     | NULL    |       |
| IC_tag_cdb_saltEBLDLGNUGR | text                | YES  |     | NULL    |       |
| UQXIKWSPXKoEq             | blob                | YES  |     | NULL    |       |
| EQZRXTDDXIoOrder          | bigint(20) unsigned | YES  |     | NULL    |       |
| cdb_saltKKFXPQDCNL        | bigint(8) unsigned  | YES  |     | NULL    |       |
```

圖 十六、終端機介面-表單匿名結構

✧ 圖十七：依照建立表單時的欄位順序計算出處方籤編號、病患姓名和藥名欄位的匿名，並嘗試查詢欄位資料，得到密文。



```
mysql> select VPGYYNHXZVoEq,QFJZWMMVOMoEq,XJJUQUGUOOoEq from table_KZVEMOJBVV;
+---------------------+-----------------------------------------------+-----------------------------------------------+
| VPGYYNHXZVoEq       | QFJZWMMVOMoEq                                 | XJJUQUGUOOoEq                                 |
+---------------------+-----------------------------------------------+-----------------------------------------------+
| 17842985374874446964 | ◆h◆◆0◆◆`◆RI2◆xL◆ ◆#B1$◆)6Y◆^LC5t◆◆◆W◆x(◆◆                    | ◆◆C◆8◆◆m◆R◆◆R◆+C◆8◆F◆>◆◆ ◆◆◆◆f◆;◆◆◆                    |
| 85376691933116491491 | ◆J4Y$◆◆;g◆◆_◆◆◆◆,n◆◆Gt◆◆\◆◆i◆◆◆je◆!&$◆◆8                    | ◆2◆E◆z百b◆◆XPI(◆◆◆◆y◆=◆KZ|◆]◆◆jrL{G>i◆◆                    |
| 87558651317435802 52 | ◆◆'◆L                                                         |                                                |
| ◆◆,◆◆$◆_qc{9n) ◆Q◆◆◆◆◆◆◆0◆◆◆                                 | k◆◆[_TgP◆◆V◆◆@◆jo◆◆◆◆e5◆K◆◆                                  | ◆[38◆7W◆◆9◆C◆zv◆                                  |
| 18114644104511311968 | G◆◆◆(M◆]▒◆ ◆P◆◆◆#S◆◆ 9◆◆◆M◆◆(◆◆*◆Fr◆◆Y                      | ◆QT◆◆,1◆◆JeZ@◆◆Hy◆M◆◆v◆l◆3◆◆◆n◆◆Q&◆◆◆◆=MN◆;                  |
| 10007555908548109350 | ◆]◆3hg◆◆◆◆gO◆b▒D/M◆◆◆l◆◆◆◆◆c◆◆                               | .◆◆◆◆:◆:◆◆xy"◆ \◆(^◆◆)◆?◆a◆◆◆◆G◆◆◆◆M                    |
| 15115734735571657763 | l◆◆◆ea&S,◆▒◆`◆-◆◆◆◆◆◆◆◆◆e[◆{◆-◆◆ksh7징^◆◆                    | ◆SV◆                                           |
| ◆(, ◆L◆◆:◆w◆B◆◆◆!◆◆b◆◆j◆z◆pn◆◆G◆                             |                                                |                                                |
| ◆                                                           |                                                |                                                |
| 17934891892037555773 | B◆p◆R◆n&◆z<◆<◆◆~◆◆◆G◆◆F+◆d-◆寮!GyC◆◆◆j                       | 7◆◆5◆@@◆f◆◆◆!N◆◆◆m◆S◆◆\◆r◆Yv◆Jeg7tk◆                    |
| 58580313170000340915 | ◆◆yw""$◆-]r{--u◆◆Q&◆◆(◆◆;▒◆◆Y◆b_◆◆◆                          | ◆I◆(◆c◆Uv◆fO◆8◆◆(@7◆g◆◆ `◆◆◆[)◆◆5◆◆X◆◆P◆LDw#                |
| 225450479066245043 | ◆4h◆◆◆36◆S◆◆EaJ◆jb◆3◆◆Xu◆ ◆AK>◆=sZ◆◆h◆#9◆◆◆                  | ◆_◆◆k◆◆◆◆◆◆AO◆◆Nj◆Id◆X◆◆H-G◆3◆◆◆◆◆                    |
| ◆U!u◆E'g◆&◆◆◆2dO◆◆7◆◆◆\◆m◆◆◆,◆◆U                            | |9                                            | /◆K[d◆T                                        |
| ◆&◆◆W=◆◆Fe◆◆lAg◆◆◆◆◆BE`◆C◆◆◆◆r◆:◆sQ◆◆E◆◆◆                   |                                               | | w◆7                                          |
| 62342008255516268948 | ◆◆◆[ob◆@◆◆B◆$3D◆!◆◆◆n◆◆v◆A◆◆/◆◆naa◆◆B◆Nl◆P                  | | ◆8◆◆◆[l      6◆8S◆◆◆ ◆◆◆◆盲H!◆Fn◆◆1◆◆H◆]◆ ◆l◆◆#          |
| 96418527753658751995 | ◆◆◆◆E-▒◆|◆`$m|!!◆ <◆-kFe9◆◆e[$@"◆◆lB@                      | ◆◆◆◆fe5◆Pd◆◆◆◆▒M◆◆)◆◆ 3◆◆◆◆&◆!◆◆(r:3◆w◆                  |
| 15431987629964982413 | ◆&◆◆◆|"◆◆◆<◆◆◆ 43◆◆◆◆2/◆{◆◆d◆l:4-'U◆3                      | \◆Db◆w◆◆◆◆◆◆H◆◆◆!◆       ◆V◆◆7v~◆L◆◆E◆▒              |
| 160487007505016317 | ◆L$◆◆                                                         |                                                |
| )◆"                                                         |                                                |                                                |
| ◆?{◆vL◆/uG|◆N◆◆teP◆◆4#◆2◆◆                                  | ◆;◆x|nnR◆◆,M◆◆Qj◆K&◆E◆V◆◆◆>/l◆s@◆_◆◆◆◆                      |                                                |
| 18368592892955635625 | ◆◆71b◆:◆◆◆◆O◆Tv◆M◆◆◆%◆ZL◆v◆◆◆▒#Z◆9◆                        | ◆b◆◆◆*◆◆◆&ÔD◆◆H◆◆O      ◆◆<:;9J\<s◆k0T◆◆L◆◆◆              |
| 14210332259856004037 | ◆j◆Z◆O◆◆U/YM2◆◆       ◆◆S◆λS◆IM◆◆cfR◆л                      | j◆◆◆LD▒◆◆9*J'◆S◆b◆◆PA◆a*>◆h◆b◆◆◆-◆m_(◆y                  |
| ◆◆O▒P                 | ◆◆u脸E◆◆◆◆◆◆[J{◆◆◆◆◆UT◆z◆_◆k                                |                                                |
| ◆]D-◆◆◆◆▒◆)◆/◆◆◆◆◆EL|◆F-◆O▒◆◆Us◆ ◆                          |                                                |
| 68270575802642861 59 | ◆<W◆/◆!?◆|◆>◆◆w+$◆◆>◆◆◆6◆◆#◆◆◆◆_◆+E◆>◆◆2                    | $◆◆◆◆Zelz◆◆v◆◆◆.◆]{&◆PÊ◆◆◆_◆◆Zf◆◆                    |
| 84140751066583100 24 | m◆◆:A◆<◆D◆◆%◆J◆◆m◆P▒◆◆]◆bz◆◆O▒◆◆,u◆◆◆N+◆                    | ◆◆◆Y◆◆9?◆◆◆◆X◆Ô◆ ◆ ◆  ◆◆p◆◆C.8◆◆A◆◆◆◆◆=|◆                |
| 16982558429470180100 | *◆7g◆◆W◆j◆F8◆◆◆E◆◆P◆a◆b9▒◆◆◆MIA◆◆N "c                       | F◆/◆◆D◆,`Qx◆:◆◆ 「◆#◆◆jt◆◆◆d◆◆J◆|◆lGu◆C                    |
| f◆\B◆GwePdL6G:@w◆◆◆◆◆+p|p◆m                 |◆◆Z5◆◆ i*8'◆3◆O.o▒◆◆/◆                    | ◆◆◆B◆a:xx3n                                    |
| 11294580125508719354 | TAV3◆p=◆+◆F◆◆◆:U◆◆◆v◆D◆◆◆'VN◆◆◆y◆\◆5◆◆t                     | E◆◆◆N◆b◆◆Ioo◆Y◆◆DM◆◆◆◆◆◆◆◆_G-q◆&)◆z,G◆H◆◆                |
| 57625864678479722 43 | U◆◆{   ◆V!9◆T);i◆◆B◆M◆◆◆(S◆r◆◆◆l◆O◆@◆- ◆◆                  | ◆◆d◆◆◆\◆ ◆xm◆+}4w◆kf◆(◆◆◆=◆V◆N◆◆%%◆◆O                  |
| V◆K◆J◆◆JHt◆[◆qs◆}U◆#:j◆i◆◆d◆!]6◆◆◆]                        |                                               | `◆.◆◆U{f◆◆◆AA◆◆◆◆y◆#◆◆k◆◆◆◆``                    |
| ◆k:◆◆                                                       |                                                |                                                |
| 16009461036756627576 | ◆◆◆◆◆Q@◆%uB◆_◆◆◆◆抵XW◆◆-◆kP◆◆Y$`◆ ◆x◆W                     | ◆u◆◆◆◆◆X◆W◆◆)◆◆aY◆◆◆|_h/`◆◆u◆◆\◆{◆r                    |
| 87349676671400505 14 | rn◆^◆7|◆◆_0Q◆Viy◆t`◆&>D$◆◆rS◆DF'3◆V◆◆y◆                    | E◆:◆g◆◆◆◆◆}◆,◆<pM◆?F◆'5▒IO◆K◆}~◆◆◆F◆                    |
| 80801319336908682 38 | OwqeF_5◆◆l◆5"◆◆◆°z底9◆◆E◆◆l◆ ◆W◆3J◆>D0◆                     | dl ◆s◆TC◆'◆◆Sp◆◆◆{g◆◆◆%◆m,◆=◆◆S◆◆f◆}                    |
| 85817922295223675 4 | ◆◆WL◆7a}◆◆◆L~◆D>◆◆8◆,Y◆L1◆u◆◆◆◆8▒s                          | ◆◆◆ghT^◆L#◆◆QmEs◆u◆Gpj[@◆◆Q◆◆_◆#E◆"◆◆                  |
| 17292330857122740412 | ◆m◆◆|◆◆◆◆.◆◆p◆◆◆3k◆◆>8◆◆◆(◆◆~G◆◆◆◆    ◆◆                  | ?◆\hg                                          |
| ◆◆檐tW◆s*◆(◆◆◆K ◆9C◆=K◆◆◆9C3j◆                             |                                                |                                                |
| 85405865558976142508 | ◆Vs/_◆-◆x◆◆BW◆\eu7◆◆`◆◆◆◆j◆s2+_Ay;◆Sr◆◆k◆                  | ◆Y◆s◆◆◆/◆◆◆qqj◆◆◆◆U◆◆yG◆◆m◆◆◆◆M◆99◆◆◆vF^Q                |
| 12406507382003314 05 | ◆/◆◆$h◆Aec:5a◆◆q◆◆n◆◆?◆G◆◆◆`~8◆h◆◆G    e◆                  | ◆◆'P◆◆◆W]◆v◆B◆◆◆◆lA◆◆◆M=◆◆D?◆^9◆@wq◆◆G                  |
```

圖 十七、終端機介面-查詢密文資料

✧ 圖十八：修改藥名欄位的第二筆資料。

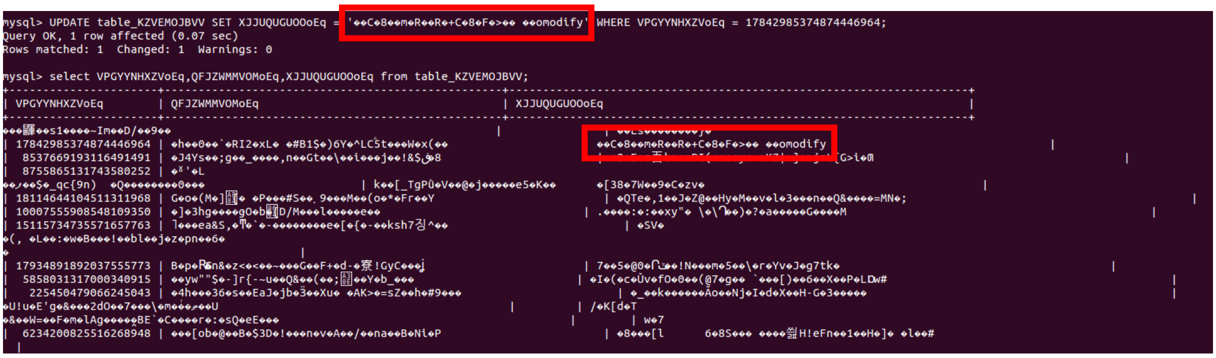


圖 十八、終端機介面-竄改密文資料(text)

✧ 圖十九：修改處方籤編號的第三筆資料

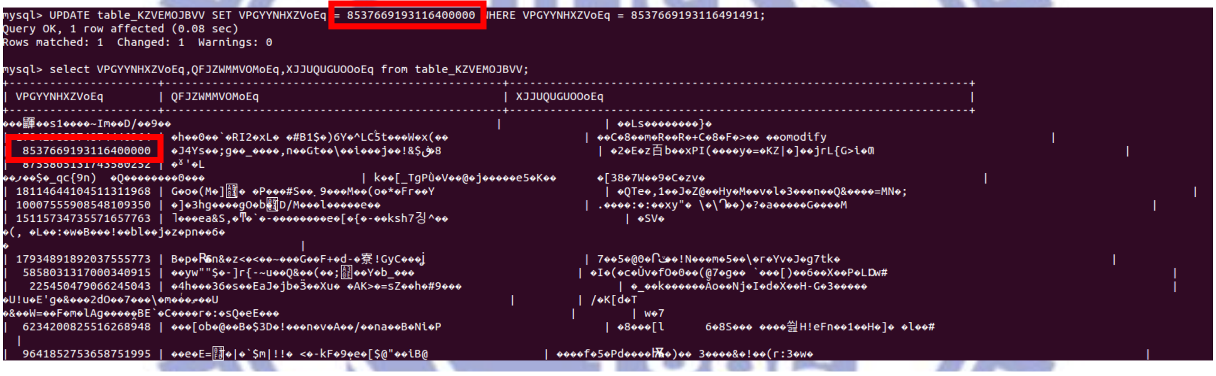


圖 十九、終端機介面-竄改密文資料(int)

✧ 圖二十：竄改密文資料後，透過代理伺服器連入 MySQL，查詢處方籤表單中的處方籤編號、病患編號、病患姓名和藥名，發生解密錯誤。

圖 二十、終端機介面-解密失敗

✧ 圖二十一：自代理伺服器的訊息，得到竄改後的密文並沒辦法通過 RND 層的解密（密文格式不符）。

圖 二十一、終端機介面-系統偵測密文格式不符

✧ 圖二十二：透過代理伺服器連入 MySQL，利用 CryptDB 提供指令剝除藥名欄位的 RND 層得到 DET 層密文。

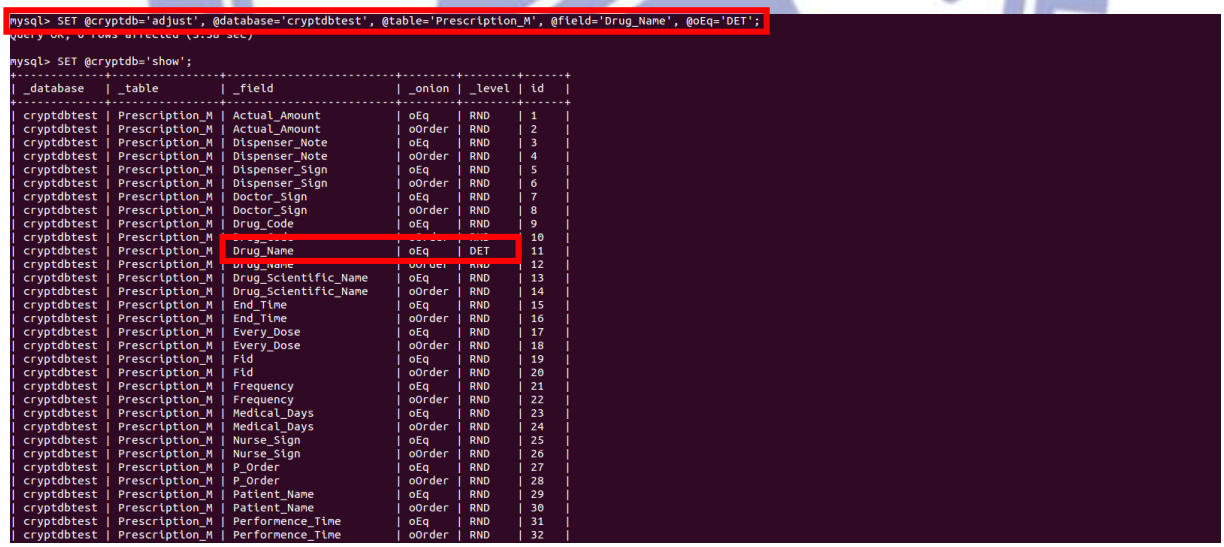


圖 二十二、終端機介面-調整洋蔥加密法加密結構(至 DET 層)

✧ 圖二十三：再一次透過代理伺服器連入 MySQL，查詢處方籤表單中的處方籤編號、病患編號、病患姓名和藥名；觀察代理伺服器的訊息：未遭竄改的欄位可以通過驗證，遭到竄改的欄位則無法通過驗證。

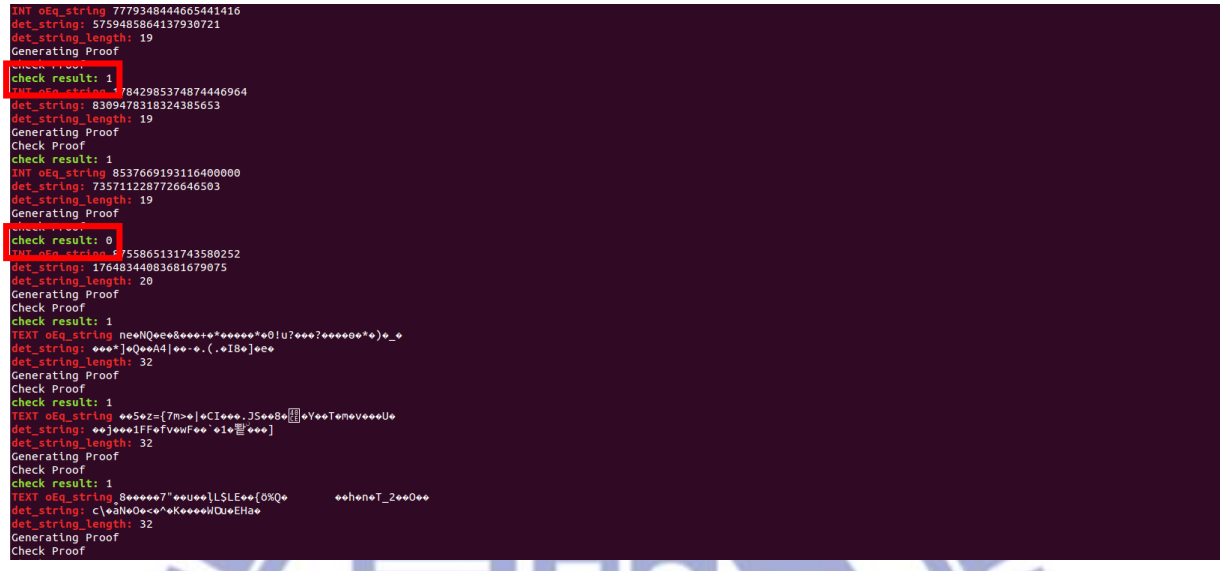


圖 二十三、終端機介面-完整性檢測 DET 層密文資料

✧ 圖二十四：獨立來看這筆密文資料是剛剛竄改的處方籤編號欄位第三筆資料。

圖 二十四、終端機介面-遭竄改的 DET 層密文資料無法通過完整性檢測

✧ 圖二十五：查詢之後回傳之後的訊息，通過完整性檢測的資料會解密，反之則會呈現完整性檢測失敗的警告。

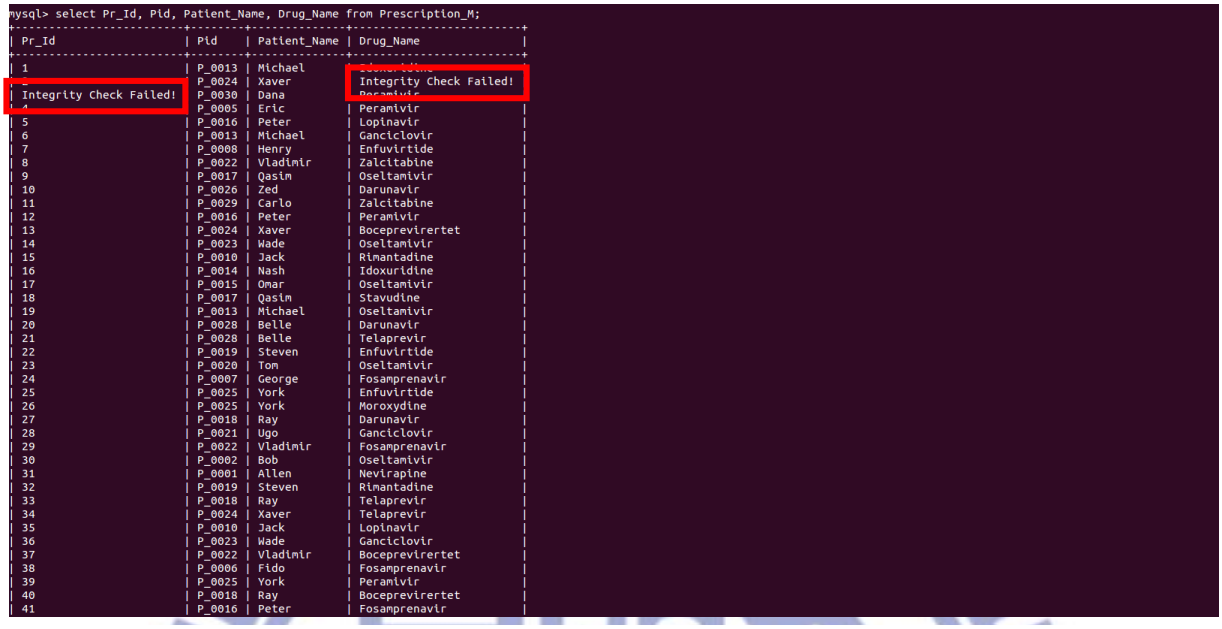


圖 二十五、終端機介面-查詢資料，發送警訊通知使用者資料未通過完整性檢測

✧ 圖二十六：改以條件查詢處方籤表單中的處方籤編號、病患編號、病患姓名和藥名，設定篩選條件是處方籤編號小於 5 的資料；仍然可以驗證我們的完整性檢測機制。

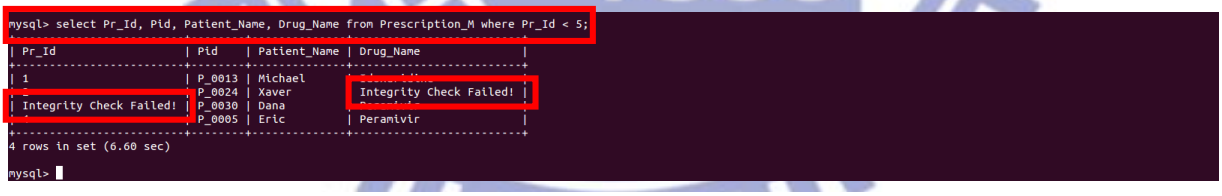


圖 二十六、終端機介面-條件查詢資料，發送警訊通知使用者資料未通過完整性檢測

5.4.3 使用者介面

高使用度(usability)的使用者介面可以增加使用者對於系統的接受度；使用者在操作時，依照以往相關介面的操作經驗，可以快速地找到方法、執行希望完成的任務。我們根據[3][19][20]等學者提供的原則與方法，設計我們的使用者介面，希望可以讓使用者直覺地操作我們的系統。

設計、實作介面期間，我們經歷四次測試，每次邀請兩至三位測試者透過介面操作我們的系統。測試過程：

1） 描述我們系統 SDEMR 的目的。
2） 給予使用者測試的帳號、密碼以及時間內希望他們完成的任務。(例如：自登入頁面開始計時 10 分鐘，完成查詢住院記錄的資訊。)
3） 觀察並且記錄使用者操作任務期間的行為。
4） 測試結束，詢問並且記錄使用者的回饋。
5） 根據 3 和 4 兩點的記錄，重新修改頁面，並規劃下一次的測試。

如以下圖示，我們提供網頁使用者介面，取代指令命令模式，降低本系統使用者的操作門檻。

- **一般使用者—病患**
  - 圖二十七：頁面的上方由左至右是我們系統的縮寫(SDEMR)、以及系統的簡單描述。頁面中央可以很容易看到登入所需要的資訊，使用者 Dana 透過輸入帳號、密碼以及選定的使用者身份登入。

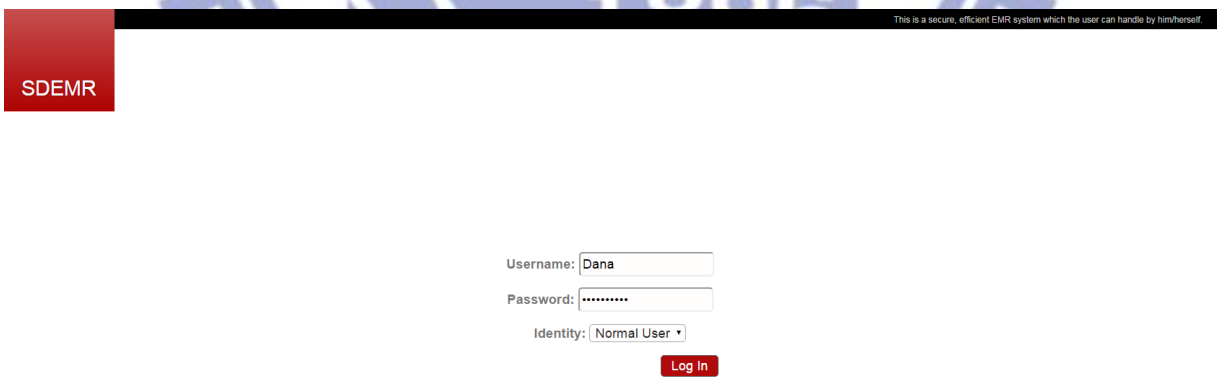


圖 二十七、使用者介面--一般使用者登入

✧ 圖二十八：Dana 登入後，頁面在右上方呈現歡迎訊息、使用者編號(User ID)以及登出(Logout)的選項；訊息下方可以看到我們系統介面的主要功能導覽列；頁面中央則是有使用者不同表單資料的頁面標籤，首先呈現個人基本資料(Patient Basic Data)。

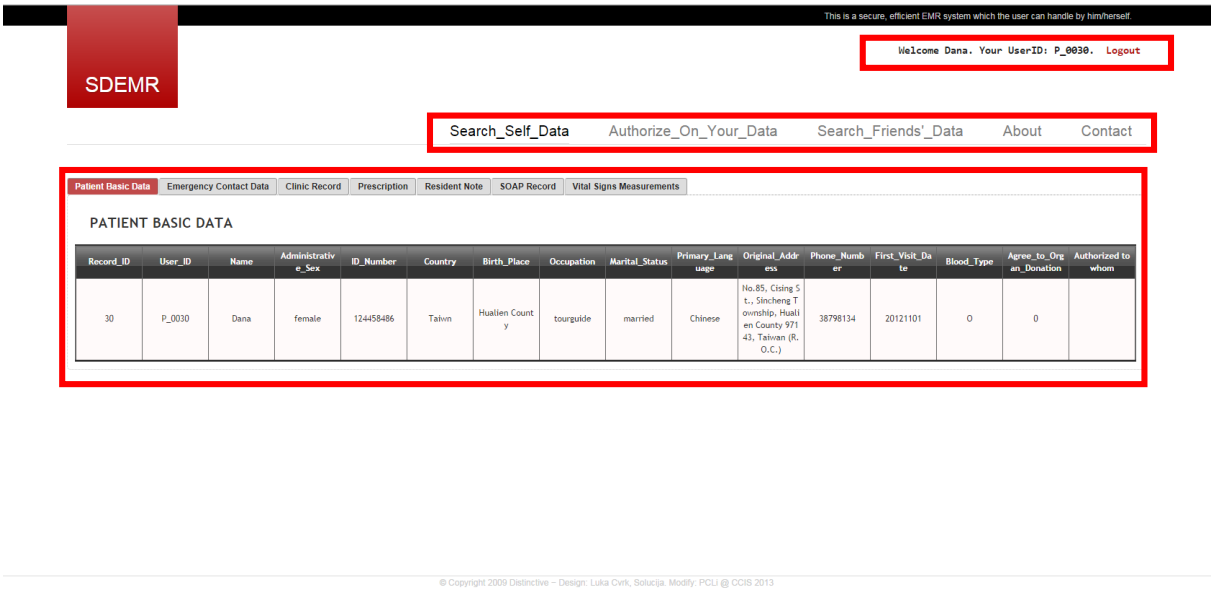


圖 二十八、使用者介面–一般使用者歡迎訊息、基本資料頁面

✧ 圖二十九：自頁面標籤選取住院記錄(Resident Note)以查看資料。

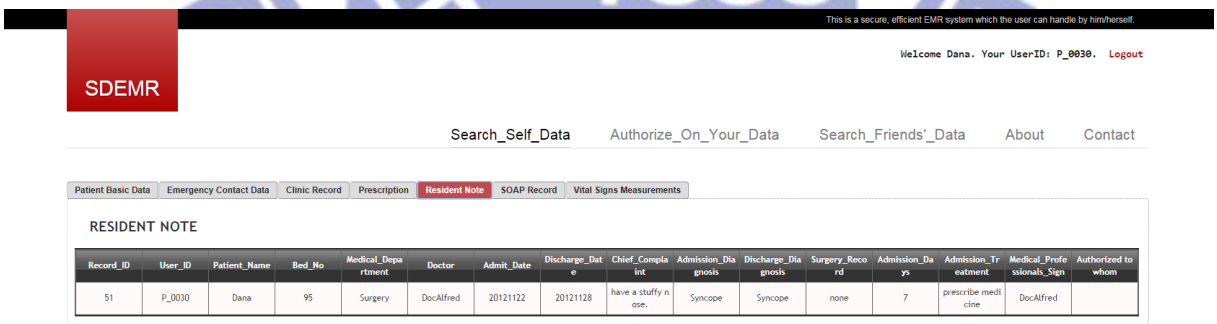
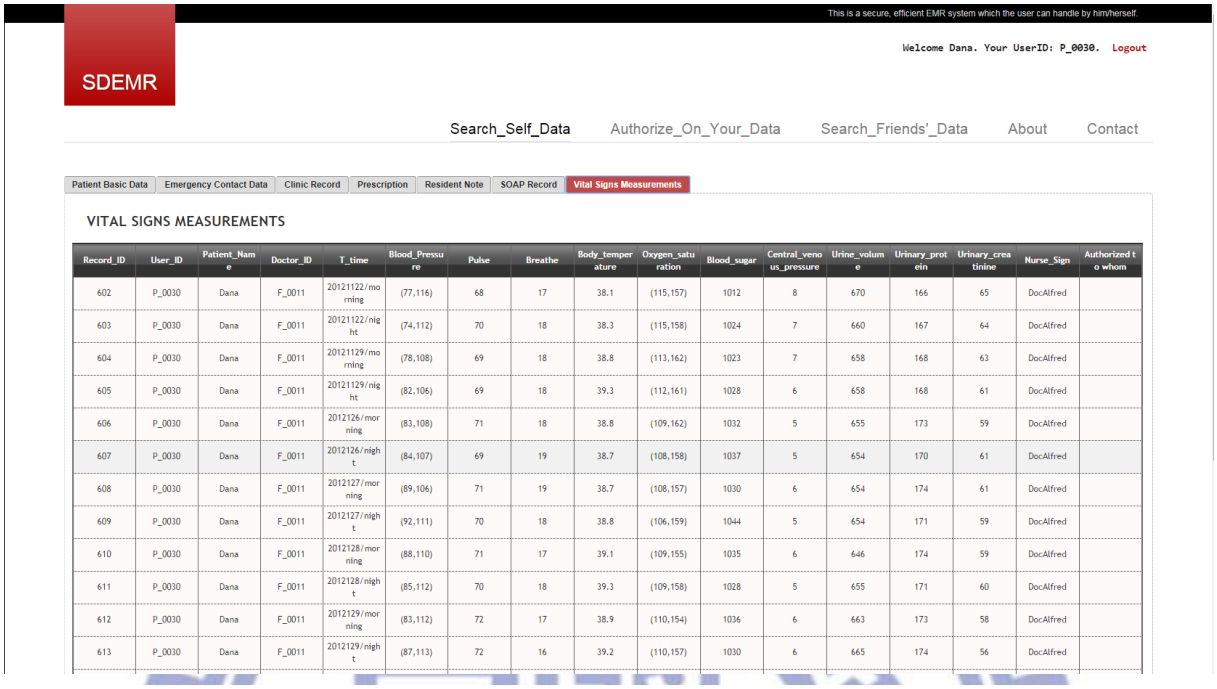


圖 二十九、使用者介面–一般使用者住院記錄

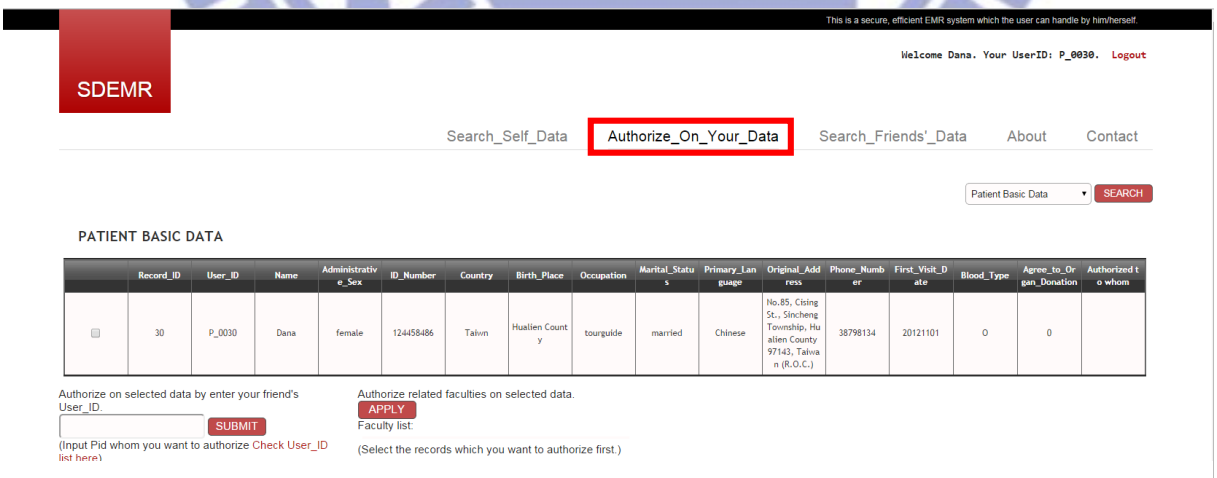
✧ 圖三十：選取生命徵象測量記錄(Vital Signs Measurements)，以查看住院期間按時測量的生理數據。



Record_ID	User_ID	Patient_Name	Doctor_ID	T_time	Blood_Pressure	Pulse	Breathe	Body_temperature	Oxygen_saturation	Blood_sugar	Central_venous_pressure	Urine_volume	Urinary_protein	Urinary_creatinine	Nurse_Sign	Authorized to whom
602	P_0030	Dana	F_0011	20121122/morning	(77,116)	68	17	38.1	(115,157)	1012	8	670	166	65	DocAlfred	
603	P_0030	Dana	F_0011	20121122/night	(74,112)	70	18	38.3	(115,158)	1024	7	660	167	64	DocAlfred	
604	P_0030	Dana	F_0011	20121129/morning	(78,108)	69	18	38.8	(113,162)	1023	7	658	168	63	DocAlfred	
605	P_0030	Dana	F_0011	20121129/night	(82,106)	69	18	39.3	(112,161)	1028	6	658	168	61	DocAlfred	
606	P_0030	Dana	F_0011	2012126/morning	(83,108)	71	18	38.8	(109,162)	1032	5	655	173	59	DocAlfred	
607	P_0030	Dana	F_0011	2012126/night	(84,107)	69	19	38.7	(108,158)	1037	5	654	170	61	DocAlfred	
608	P_0030	Dana	F_0011	2012127/morning	(89,106)	71	19	38.7	(108,157)	1030	6	654	174	61	DocAlfred	
609	P_0030	Dana	F_0011	2012127/night	(92,111)	70	18	38.8	(106,159)	1044	5	654	171	59	DocAlfred	
610	P_0030	Dana	F_0011	2012128/morning	(88,110)	71	17	39.1	(109,155)	1035	6	646	174	59	DocAlfred	
611	P_0030	Dana	F_0011	2012128/night	(85,112)	70	18	39.3	(109,158)	1028	5	655	171	60	DocAlfred	
612	P_0030	Dana	F_0011	2012129/morning	(83,112)	72	17	38.9	(110,154)	1036	6	663	173	58	DocAlfred	
613	P_0030	Dana	F_0011	2012129/night	(87,113)	72	16	39.2	(110,157)	1030	6	665	174	56	DocAlfred	

圖 三十、使用者介面--一般使用者生命徵象記錄

✧ 圖三十一：自導覽列，連結我們的系統未來將會發展的"個人資料授權"頁面(Authorize_On_Your_Data page)。



	Record_ID	User_ID	Name	Administrative_Sex	ID_Number	Country	Birth_Place	Occupation	Marital_Status	Primary_Language	Original_Address	Phone_Number	First_Visit_Date	Blood_Type	Agree_to_Organ_Donation	Authorized to whom
☐	30	P_0030	Dana	female	124458486	Taiwn	Hualien County	tourguide	married	Chinese	No.85, Cising St., Sincheng Township, Hualien County 97143, Taiwan (R.O.C.)	38798134	20121101	O	O	

Authorize on selected data by enter your friend's User_ID.
SUBMIT
(Input Pid whom you want to authorize Check User_ID list here)

Authorize related faculties on selected data.
APPLY
Faculty list:
(Select the records which you want to authorize first.)

圖 三十一、使用者介面--一般使用者資料授權頁面

✧ 圖三十二：資料授權操作示意，勾選欲授權的醫療資料，填入欲授權的使用者編號；或是直接授權系統預設的相關醫事人員。

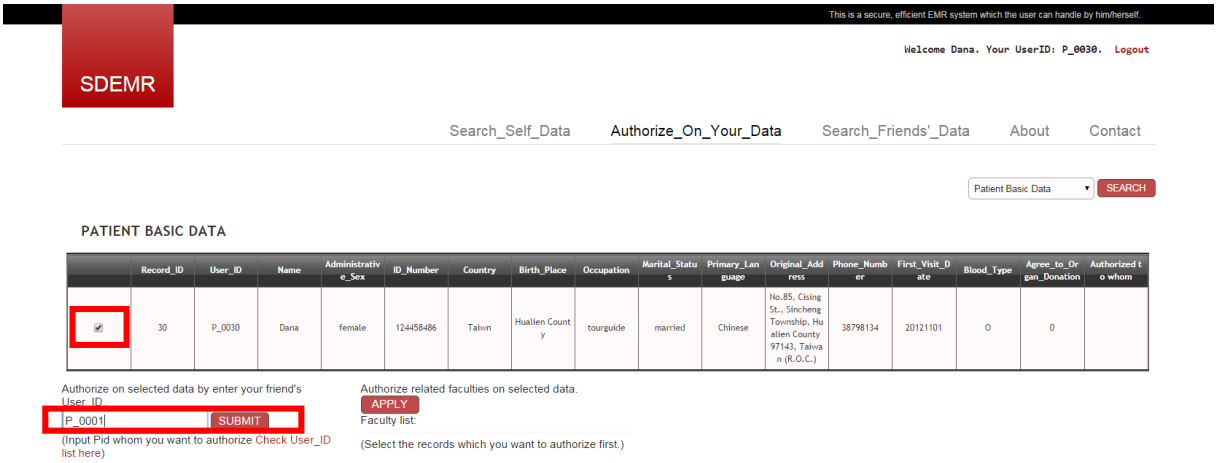


圖 三十二、使用者介面--一般使用者資料授權操作概念

✧ 圖三十三：備有使用者姓名和編號對應表，以供查詢。

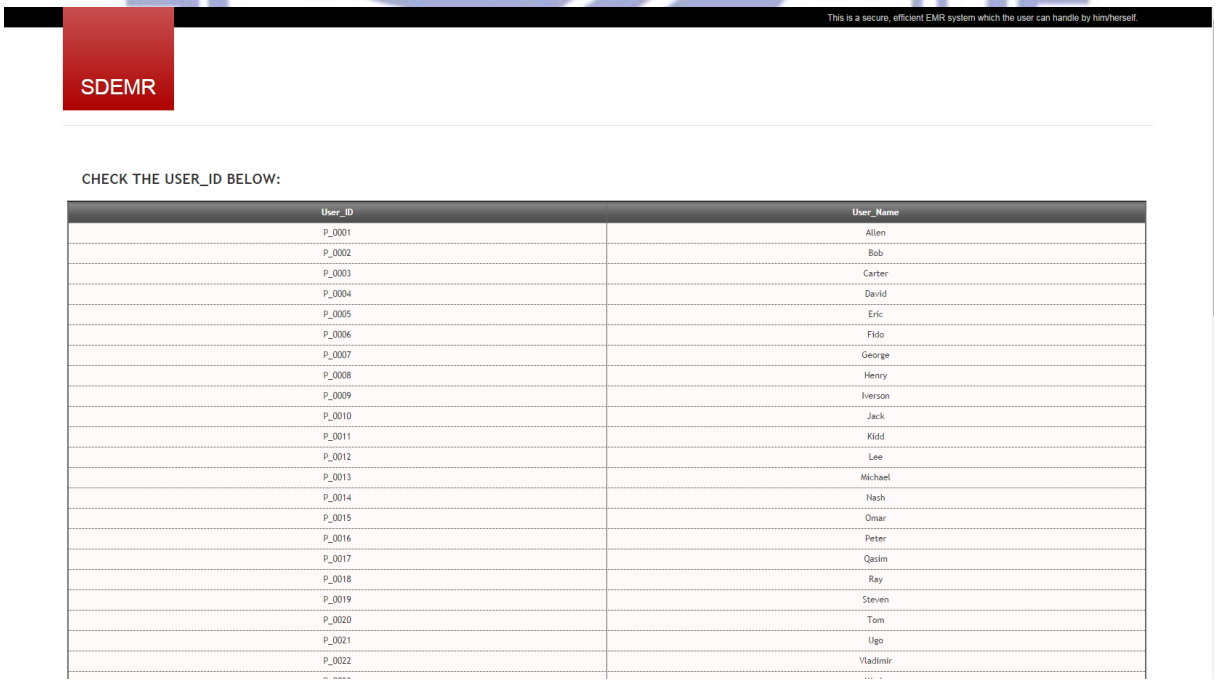


圖 三十三、使用者介面--一般使用者姓名、編號對照表

✧ 圖三十四：自導覽列，連結我們的系統未來將會發展的”查詢友人的資料”頁面(Search_Friends’_Data page)。可以查詢友人經過授權功能分享的資料。

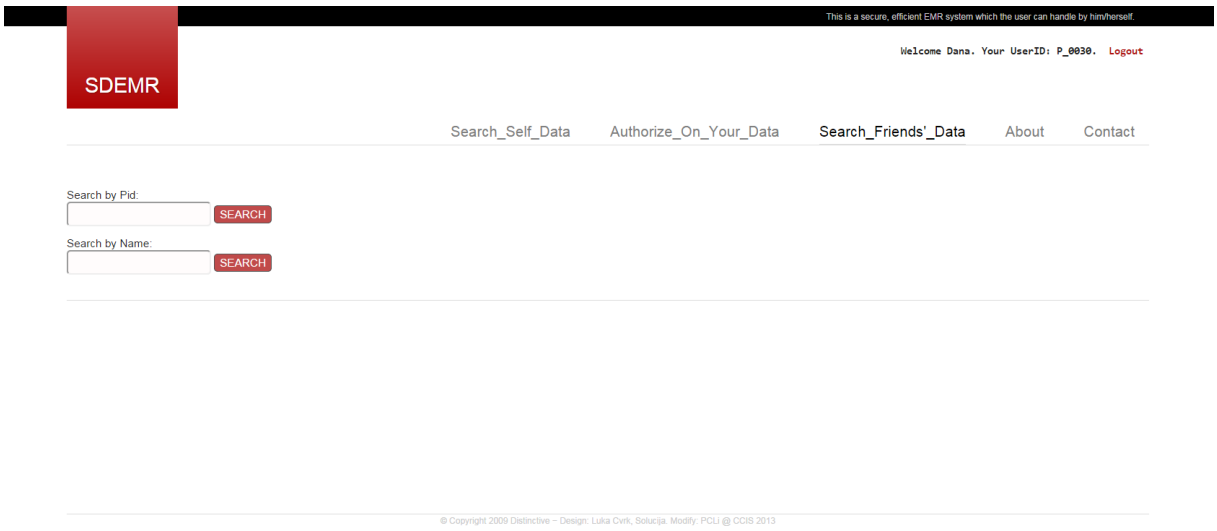


圖 三十四、使用者介面--一般使用者搜尋友人資訊頁面

➢ **醫事人員**

✧ 圖三十五：登入頁面，醫師 Aaron (DocAaron)輸入帳號、密碼並選取登入身分—醫事人員(Faculty)。

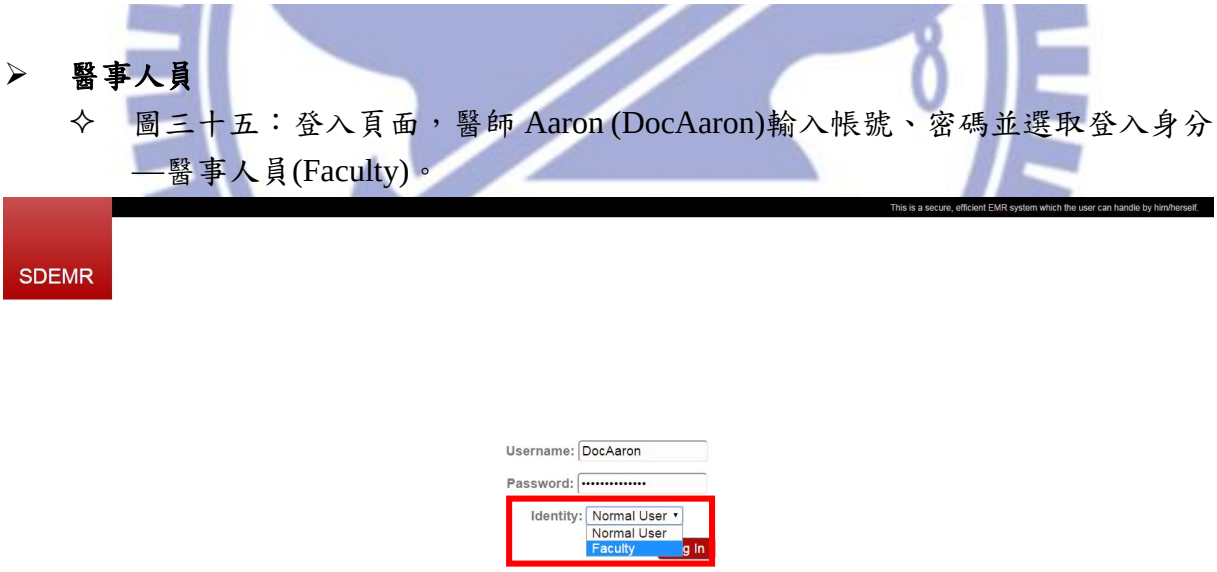


圖 三十五、使用者介面-醫事人員登入

✧ 圖三十六：醫師 Aaron 登入後的歡迎訊息、頁面導覽列以及個人基本資料頁面。

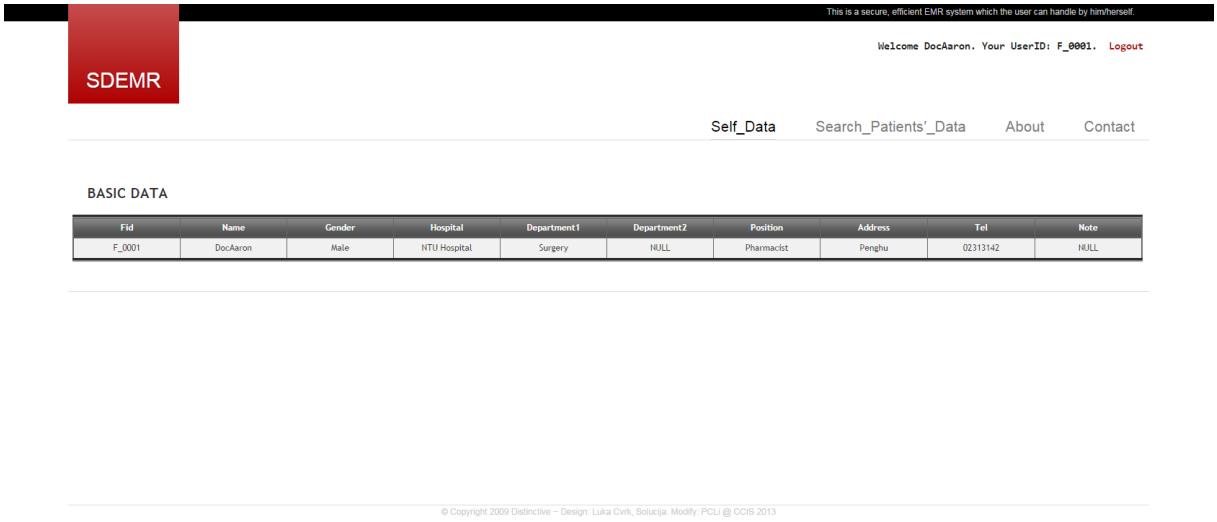


圖 三十六、使用者介面-醫事人員歡迎訊息、基本資料頁面

✧ 圖三十七：自導覽列，連結至我們的系統未來將會發展的"查詢過去看診病患資料"頁面(Search_Patirnts'_Data page)，可以透過搜尋病患的姓名、編號來查詢相關資料。或是直接列出所有得到授權的病患資料。

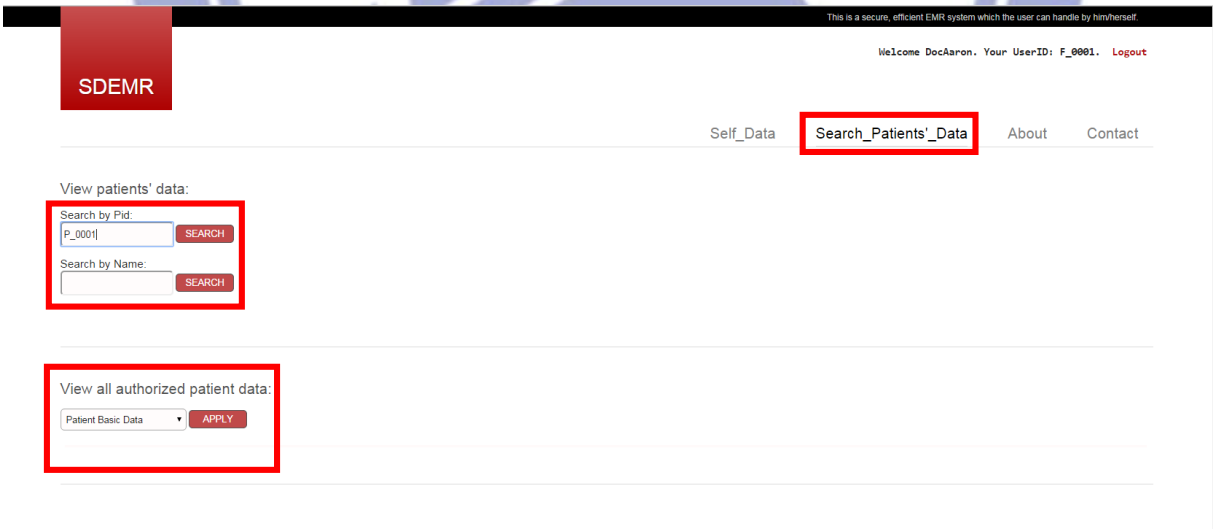


圖 三十七、使用者介面-醫事人員搜尋病患資料頁面與操作概念

第 6 章 結果與討論

我們設計並且開發一個安全且支援資料完整性檢測、MySQL 常見操作的醫療資料庫系統。

我們以 CryptDB 元件為基礎，利用洋蔥加密法，保證醫療資料的私密性，同時支援 MySQL 資料庫的常見操作。為了使整個系統更符合醫療資料庫上線至雲端儲存裝置的需求，我們建立資料完整性檢測的機制：在資料被使用之前，檢測資料的完整性，以保證在雲端儲存裝置的資料有被正確地儲存。

我們也參照行政院衛生部推動的電子病歷管理系統中提供的單張基本格式，選取八張表單、建立五十名病患和五十名醫事人員的醫療資料以模擬本系統的功能。另外，我們設計了網頁使用者介面取代命令模式；使用者可以透過網頁操作直覺地操作我們的系統。

目前本系統的完整性檢測機制依照欄位(field)建置，雖然可以完全確保資料的完整性，也保有 CryptDB 的特點，但有完整性標籤過大的問題。如何維持現有的系統優勢，並有效地降低完整性標籤的儲存空間，實為我們的系統未來可以再修正的方向。

另外，受限於 CryptDB 元件支援的資料型態，資料庫中各表單的資料型態均為 int 及 text。雖然對於本系統模擬小型醫療資料庫已經符合使用需求，但是為了更符合一般 MySQL 資料庫操作，在未來應該繼續開發支援其他資料型態，如：datetime、float。

參考文獻

[1] R.A. Popa, C.M.S. Redfield, N. Zeldovich, and H. Balakrishnan. “CryptDB: Protecting Confidentiality with Encrypted Query Processing.” In Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP 2011), pages 85-100, Cascais, Portugal, October 2011.

[2] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song. “Provable Data Possession at Untrusted Stores” In Proceedings of the 14th ACM conference on Computer and Communication Security (CCS 2007), pages 598-609, Alexandria, VA, USA, October 2007.

[3] J. Nielsen. “Usability 101: Introduction to Usability.” January, 2012. http://www.nngroup.com/articles/usability-101-introduction-to-usability/

[4] 衛生福利部，電子病歷管理系統，單張基本格式. http://emrstd.mohw.gov.tw/strdoc/DocLib/Forms/AllItems.aspx

[5] 維基百科，全民健康保險. http://zh.wikipedia.org/wiki/%E5%85%A8%E6%B0%91%E5%81%A5%E5%BA%B7%E4%BF%9D%E9%9A%AA

[6] R.A. Popa and N. Zeldovich. “Cryptographic treatment of CryptDB's Adjustable Join.” Technical Report MIT-CSAIL-TR-2012-006, Computer Science and Artificial Intelligence Laboratory, Cambridge, MA, March 2012.

[7] C. Curino, E.P.C. Jones, R.A. Popa, N. Malviya, E. Wu, S. Madden, H. Balakrishnan, and N. Zeldovich. “Relational Cloud: A Database-as-a-Service for the Cloud.” In Proceedings of the 5th Biennial Conference on Innovative Data Systems Research (CIDR 2011), Pacific Grove, CA, January 2011.

[8] K. D. Bowers, A. Juels, and A. Oprea. “Proofs of Retrievability: Theory and Implementation.” In Proceedings of the 2009 ACM workshop on Cloud Computing Security (CCSW 2009), pages 43-54, Hyatt Regency Chicago, Chicago, IL, USA, November 2009.

[9] C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia, “Dynamic Provable Data Possession.” In Proceedings of the 16th ACM conference on Computer and Communication Security (CCS 2009), pages 213-222, Hyatt Regency Chicago, Chicago, IL, USA, November 2009.

[10] F. Liu, D. Gu, and H. Lu. “An Improved Dynamic Provable Data Possession Model.” IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS), pages 290 – 295, Beijing, China, September 2011.

[11] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li. “Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing.” IEEE Transactions on Parallel and

Distributed Systems, volume 22, issue 5, pages 847-859, May 2011.

[12] Q. Wang, C. Wang, J. Li, K. Ren, W. Lou. ”Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing.” Computer Security – ESORICS 2009, The 14th European Symposium on Research in Computer Security, pages 355-370, September 2009.

[13] Y. Ren, J. Xu, J. Wang and J.-Uk Kim “Designated-Verifier Provable Data Possession in Public Cloud Storage.” International Journal of Security and Its Applications, volume 7, No.6 (2013), pages 11-20, July 2013.

[14] X. Liu, J. Ma, J. Xiong, T. Zhang, and Q. Li. “Personal Health Records Integrity Verification Using Attribute Based Proxy Signature in Cloud Computing.” The 6th International Conference Internet and Distributed Computing Systems (IDCS 2013), pages 238-251, Hangzhou, China, October 2013.

[15] Wikipedia, ISO/DIS 9241-11(ISO, 1997).
http://en.wikipedia.org/wiki/ISO_9241#ISO_9241-11

[16] M. Hassenzahl, A. Beau, and M Burmester. “Engineering joy.” IEEE Software, volume 18, issue 1, pages 70-76, January–February 2001.

[17] G. Lindgaard and C. Dudek. “What is this evasive beast we call user satisfaction?” Interacting with Computers, volume 15, issue 3, pages 429-452, June 2003.

[18] WAMMI.
http://www.wammi.com/

[19] J. Nielsen, “10 Usability Heuristics for User Interface Design.” January 1995.
http://www.nngroup.com/articles/ten-usability-heuristics/

[20] J. Gube. “7 Best Practices for Improving Your Website's Usability.” September 2011.
http://mashable.com/2011/09/12/website-usability-tips/

[21] 衛生福利部，電子病歷推動專區。
http://emr.mohw.gov.tw/nowProject.aspx

[22] R.A. Popa, F. H. Li, and N. Zeldovich. “An Ideal-Security Protocol for Order-Preserving Encoding.” Security and Privacy (SP), 2013 IEEE Symposium on, pages 463 – 477, Barkerly, CA, USA, May 2013.

[23] C. Wang, K. Ren, S. Yu, and Urs, K.M.R. “Achieving usable and privacy-assured similarity search over outsourced cloud data.” IEEE INFOCOM 2012, pages 451 – 459, Orlando, Florida, USA, March 2012.

[24] C. B¨osch, Q. Tang, P. Hartel, and W. Jonker. “Selective Document Retrieval from Encrypted Database.” 15th International Conference Information Security Conference (ISC 2012), pages 224-241, Passau, Germany, September 2012.

[25] D. Nunez, I. Agudo, and J. Lopez. “Integrating OpenID with proxy re-encryption to enhance privacy in cloud-based identity services.” 2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom 2012), pages 241-248, Taipei, Taiwan, December 2012.

[26] G. Ateniese, K. Fu, M. Green, and S. Hohenberger. “Improved proxy re encryption

schemes with applications to secure distributed." Journal ACM Transactions on Information and System Security (TISSEC 2006), volume 9, issue 1, pages 1-30, February 2006.

[27] Z. Hao, S. Zhong, and N. Yu. "A Privacy-Preserving Remote Data Integrity Checking Protocol with Data Dynamics and Public Verifiability." IEEE Transactions on Knowledge and Data Engineering, volume 23, issue 9, pages 1432-1437, September 2011.