國立交通大學

資訊科學與工程研究所

碩士論文

個人密碼管理系統

Personal Password Management System

研究生：謝文均

指導教授：楊　武　教授

中華民國　九十六　年七月

個 人 密 碼 管 理 系 統
Personal Password Management System

研 究 生：謝文均　　Student：Wen-Chun Hsieh

指導教授：楊　武　　Advisor：Wuu Yang

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

July 2007

Hsinchu, Taiwan, Republic of China

中華民國九十六年七月

個人密碼管理系統

學生：謝文均　　指導教授：楊　武 博士

國立交通大學資訊科學與工程研究所

摘　　要

有爆炸性成長的網際網路提供了使用者爲數眾多的服務項目，舉凡綜合性質的入口網站、打著 web 2.0 口號的 Blog、相簿、個人化入口網站、維基百科，甚至網路購物、網路 ATM、網路報稅…等等，性質五花八門；然而網路的匿名性質造成使用者必須針對不同的網路服務來註冊個別的使用者帳戶。

對於眾多網路服務的註冊需求，多數的使用者傾向使用簡單的帳號/密碼以方便記憶，不良的使用習慣使得大部分的帳號/密碼都擁有類似的形式，若是遭受網路釣魚等攻擊則會承受很大的損失；而某些例如由系統管理者所指定有著特殊形式的帳號/密碼，或是使用者鮮少登入的網路服務，又常遭使用者遺忘。對於數量不斷增長的使用者密碼的管理是一個重要的議題。

本論文提出一個能夠安全管理使用者帳戶資訊的個人密碼管理系統，利用加密演算法的混合使用來達到保護資料的高安全性；加密演算法與金鑰的使用藉由一份 Key-File 來做管理，並可將其匯出與存有帳戶資訊的 Data-File 分別保存於不同地方增加系統的安全性。

個人密碼管理系統建構在 Java 平台上，利用其跨平台的特性來達到 Portable 的需求；並設計直觀且簡易操作的 GUI 介面增加系統的親和力，User Friendly 亦是本系統強調的重點。

Personal Password Management System

Student：Wen-Chun Hsieh Advisor：Dr. Wuu Yang

Institute of Computer Science and Engineering

National Chiao Tung University

Abstract

The rapid expansion of the Internet has provided user with a divers range of services. For example, a comprehensive portal、the websites which hits “web 2.0” slogans like blog、album、personalized start page、Wikipedia; Even the online shopping、web ATM、eTAX…and so on. However, the anonymous nature of Internet makes people need to registered users of individual accounts for different network service.

To multitudinous Internet service registration demand, using simple Account/Password to facilitate memory is the majority tendency. The poor user habit makes most of the Account/Password pair have similar forms, it will result in the losses if encounter the phishing attacks. Some Account/Password pair which not frequently to use or have some specific form are forgotten easily. It's an important issue for the growing number of users password management.

This paper presents a secure user account information manager called Personal Password Management, mixed using the encryption algorithms achieves data high security. The using of encryption algorithms and secret keys is managed by a Key-File, which can exports and stores in different places with the account information Data-File to increase the system's security.

Personal Password Management System developed on Java platform, use its cross-platform characteristic to achieve the demand for Portable. It design an intuitive and easy to use GUI interface to increase system accessibility. User Friendly is the focus in the system.

致謝

研究所生涯的兩個寒暑，我要誠摯且衷心的感謝我的指導教授，楊武博士。這兩年以來老師細心且耐心的，在學生的研究上無論是靈感的啓發或是難題的解決，以及做研究的態度與方法，都詳細的給予指教。若非您的諄諄教誨，學生在兩年中是不能夠獨自摸索且順利的完成本論文的。

當然，要感謝的少不了這兩年一起奮鬥的同學們，志誠、癸夫、還有冠志。多少熬夜的夜晚，昏睡的下午，還有口試前的緊張不安，還好實驗室中有了你們的陪伴與互相鼓勵，很高興有這樣的緣份能與你們度過這充實的兩年。炒熱了我們實驗室冷冷氣氛的學弟們，奕圻、禮君、俊宇、還有帥維，也要感謝你們，不時的關心與問候讓我熬過了緊繃的時光。還有大學以來的好朋友，振哲，要特別感謝你在口試前給予的建議，讓我順利的通過這個考驗。

最後，要感謝我最愛的家人，老爸老媽、老哥、與我的女友兆嫻。謝謝你們在背後支持著我，讓我能夠心無旁騖的專心完成我的論文。感謝老爸老媽不時給予我加油打氣，以及你們無悔的支持；感謝我的女友，豐富彩色了我的碩士生活，你們是我心靈的支柱。感謝所有陪伴我走過這兩年研究生涯的人，謝謝你們。

This work was supported in part by Taiwan Information Security Center (TWISC), National Science Council under the grants NSC 95-2218-E-001-001, and NSC95-2218-E-011-015.
資通安全人才培育計畫(II)-資通安全人才培育計畫(II)之計畫編號： NSC95-2218-E-001-001 與資通安全人才培育計畫(II)-資通安全研究與教學中心(II)之計畫編號： NSC95-2218-E-011-015.

目錄

圖目錄

第 1 章
序論

密碼在生活中扮演的角色越來越重要，網際網路世界中密碼代表確認使用者身份的一把鑰匙，而我們使用 Encryption 這個「保險箱」來保護我們許多的鑰匙。如何製造一個強大的「保險箱」來保管我們的密碼，以及能夠解開它的金鑰的管理，則是本篇論文將探討的內容。

1.1 研究背景與動機

西元 2000 年當時正值 .COM 泡沫化，Bill Gates 曾說過這麼一句話：「人們都高估了這兩年網路的影響力，卻低估了它在十年後的影響力。」至今還不到十年的時間，呈現爆炸性成長的網際網路已讓此預言成真。

網路提供了人們爲數眾多且方便的服務：資訊的整合提供者如各式綜合性質的入口網站、論壇、各種主題的專門網站；打著 web 2.0 的口號者如 Blog、相簿、影片分享、維基百科、個人化入口網站；甚至各種娛樂來源的取得、拍賣網站、網路 ATM、網路報稅…等。今日人們已習慣在網路上消費購物、閱讀新聞、休閒娛樂以及分享生活瑣事，網路在不知不覺中融入了日常生活。

然而，由於網路的匿名性質，使用者必須對於不同的網路服務提供者註冊個別的使用者帳戶，以代表其在網路上的身份。對於眾多網路服務的註冊需求，多數的使用者傾向使用簡單的帳號/密碼以方便記憶；不良的使用習慣使得大部分的帳號/密碼都擁有類似的形式，進而造成某種形式 Security 上的威脅性，譬如若受到 Phishing attack 造個人資料外流，這些以相似形式註冊的帳戶資訊都有遭受威脅的可能。另外某些例如由系統管理者所指定有著特殊形式的帳號/密碼，或是使用者鮮少登入的網路服務，又常遭使用者所遺忘。不僅是上述網路服務的帳密/密碼，像是個人的銀行 Account 等等一些日常生活中的私人資料，在久未使用的情況下也很容易遺忘。

爲了避免個人帳戶資訊的遺失，最常見的做法不外乎記錄在一份文件內作爲備份，在有需要時做查詢之用。如此卻又引出兩個問題：檔案是否有足夠的

加密保護，避免私人資料外流；以及帳戶資訊的更新管理是否與備份檔案同步。

1.2 研究目標

細究人們的不良使用習慣絕大部分是由惰性所造成，無論是不適當又過於簡單的密碼形式，從不更新且隨處使用一份唯一的密碼，或是未經加密的個人資訊檔案都是如此。每一項帳號/密碼 pair 皆代表了使用者的一份個人資訊，對於存放使用者眾多帳戶資訊的檔案必須要有高度的保密性。

保護一份以Plaintext形式的檔案不受惡意使用者攻擊是沒有意義的，而一份Ciphertext的安全性又依賴著其加密金鑰的安全性[1]，因此將加密資料與金鑰存放在一起並不是明智之舉。金鑰的管理是一個重要的議題：應該存放在什麼地方？該如何保護？更新的週期多長？

本論文針對以下幾點目標，來達到安全的記錄與管理使用者的各類帳戶資訊，圖 1 - 1為其概觀：

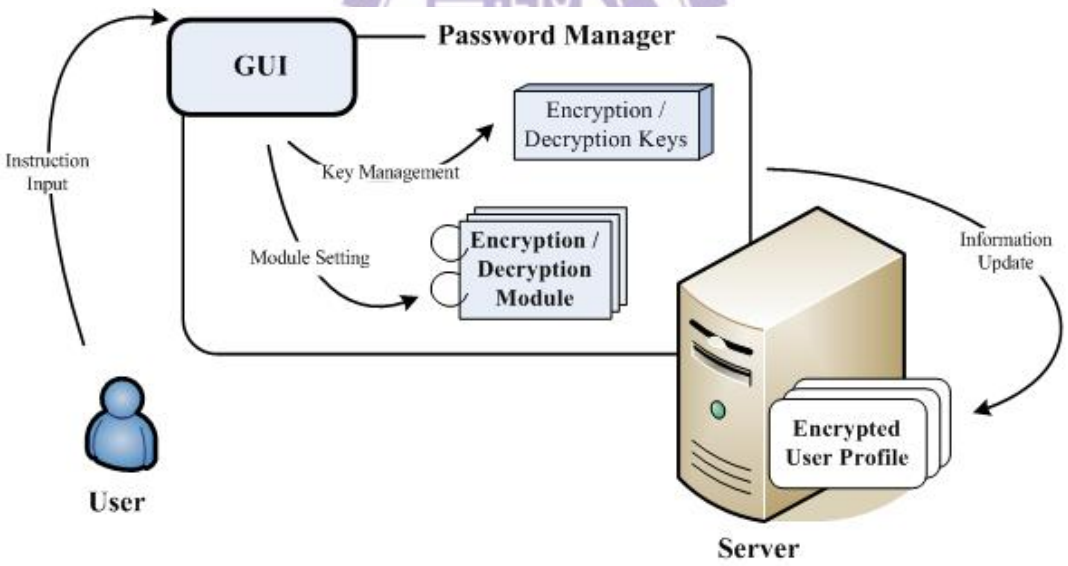


圖 1 - 1　系統概觀

- 混合使用加密演算法

  使用者能夠對於不同的欄位自行選擇加密演算法與加密金鑰，分層保護重要性不同的帳戶；並且避免同時遭到破解的可能性，達到更高的安全性

- 金鑰管理

  透過 GUI 對混合使用加密演算法的眾多金鑰做管理，獨立存放於一份

Key-File 中。使用者能夠產生多分不同的金鑰來管理帳戶資訊，並且選擇性的匯出作爲真正的“Key”，可與存有帳戶資訊的 Data-File 分別保存於不同地方來增加系統的安全性

- 簡易使用的介面

  系統最重要的就是要有 User Friendly 的使用介面，能夠幫助使用者便於記錄與維護其帳戶資訊。使系統更具親和力，對帳戶資訊有便利的編輯能力，與達到快速更新資料的目的都有正面的效果。方便易上手的介面能夠改變人們不良的使用習慣

1.3 論文架構

本論文總共分爲六個章節：第一章介紹本論文的研究背景動機，以及研究目標；第二章分析網路使用者的密碼使用習慣以及比較目前已有實作出的密碼管理系統；第三章詳述系統架構、功能需求及設計；第四章剖析個人密碼管理系統的實作細節；第五章進行系統加密與 Key Management 的測試；最後第六章則闡述系統特點，並做爲本論文的結論，以及對於本系統的未來展望。

第 2 章
相關研究

要成功防護一套系統免遭入侵，最好的方法是以攻擊者的角度來觀察系統的架構並分析其缺陷。本章對網路使用者的密碼使用習慣，Phishing 攻擊，與當前已有實作出成品類似的密碼管理系統進行分析研究。

2.1 密碼使用習慣與網路釣魚

某些 Security 方面出現的問題不是因爲系統的缺陷，而是由於不良的使用習慣所造成。台大不良牛 BBS (telnet://bbs.badcow.com.tw) 的 Security 看板曾經在 1998 年 5 月時舉辦調查網路使用者的密碼使用習慣的投票 (附錄 1)：

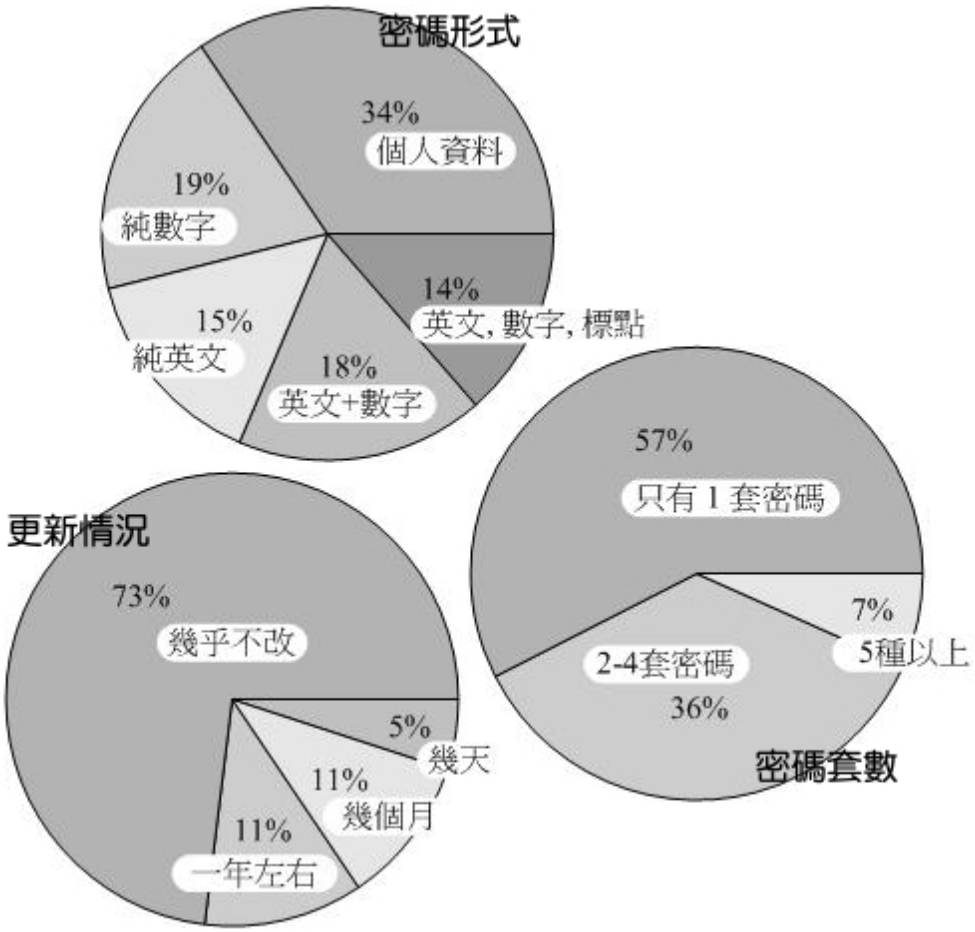


圖 2 - 1　密碼使用習慣

雖然不良牛所舉辦的這個投票有以下兩項缺點：每人可投票數過多 (八票)、以及許多選項類似，性質重複；不過仍然具有相當的參考價值。藉由圖 2 - 1對於各項目加以分類比較後我們可以清楚發現大部分的使用者密碼都有著由個人資料所組成的簡單形式，以及甚至是幾乎從不更新、且一套唯一的萬年密碼。

今日的資訊安全已經不像往日單純，即使系統或是檔案資訊都有完善的保護，新興的詐騙手段與社交工程的攻擊都會讓個人資料輕易外洩。最好的例子就是網路釣魚 Phishing。

eBay、Paypal、Citibank是三個最常被利用的目標[2]，攻擊者僞稱自己是上述機構，僞造Email以資料過期、資料無效、或是以安全性理由，誘騙受害者至攻擊者所僞造的網頁進行身份認證，來騙取個人機密資料。

一般的Phishing attack仍然無法騙取小心謹慎的使用者資料，但有較Phishing更進一步的攻擊手法“網址嫁接 (Pharming)”[3]，攻擊者藉由入侵DNS Server，竄改DNS Server cache裡的Domain IP mapping，因此又被稱爲DNS下毒(DNS Poisoning)。當使用者透過此Server查詢網址時，則被導入由攻擊者僞造的網站，若使用者進而進行登入動作，個人資料就會被盜取。Pharming除了對DNS做手腳外，另外的手法是先騙取使用者下載%SYSTEMROOT%/system32/drivers/etc/hosts檔案 (Windows User)，這分檔案中就包含了Domain Name與IP對應記錄，使用者瀏覽網頁時會優先查詢此檔中的記錄，進而達成騙取資料的目的。

不良的密碼使用習慣若遇到 Phishing 的攻擊造成個人帳戶資料外洩，其餘以類似形式帳戶/密碼註冊的服務極有可能同時遭到竊取。

2.2 類似的密碼管理系統

在網路上可以找到類似本系統的密碼管理程式，本節則挑選UMP[4]、KisKis[5]、Password Safe[6]、PassReminder[7]、KeePass[8]等其中設計較佳的五支程式來分析優缺特色。

2.2.1 UPM

Universal Password Manager (UPM)，使用 Java language 開發，能夠在不同

OS 環境下正常運作，database 使用 AES algorithm with 256 bits key length 加密。

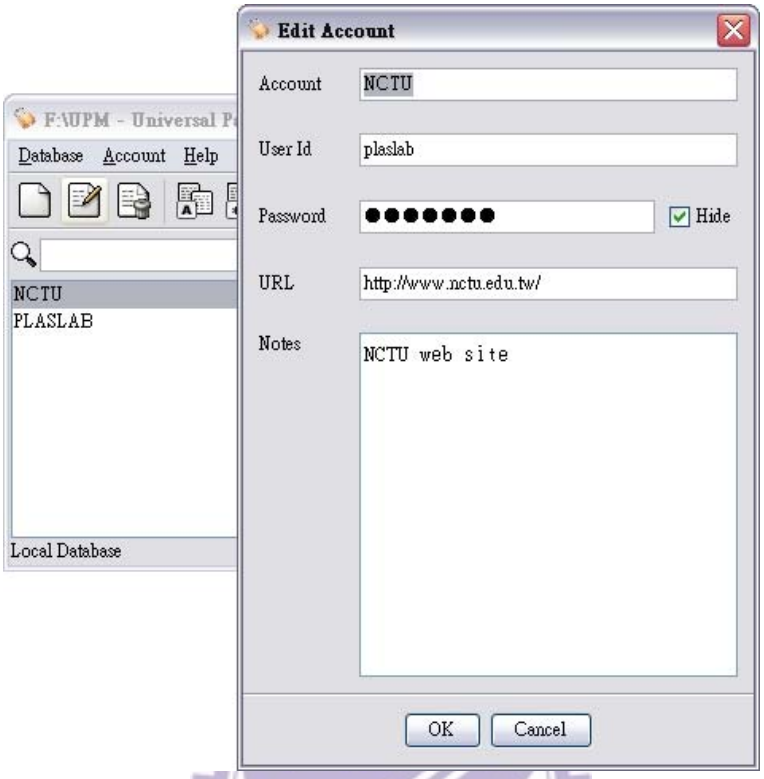


圖 2 - 2　　Universal Password Manager

在圖 2 - 2中可以看到UPM的使用畫面，UPM的特色爲簡易使用，簡潔的GUI讓使用者容易上手，且提供database export機制方便匯出使用者帳戶資訊；其缺點爲提供功能少，僅對於使用者帳戶資訊做簡易記錄的動作，且對於帳戶的編排使用list的方式全部置放於視窗下半方格，沒有提供分類管理的功能讓使用者帳戶看來雜亂。

2.2.2　KisKis

Keep It Secret! Keep It Safe! (KisKis)，亦是使用 Java language 寫成，提供跨平台使用的特性，database 使用 XML 的檔案格式，而在 database 加密方面則讓使用者選擇使用 AES with 128 bits key length、Blowfigh、CAST5 三種加密法之一。

圖 2 - 3爲KisKis的執行畫面，可以清楚看到使用了Java Swing的GUI Look and Feel。不同於UPM使用list方式，KisKis使用樹狀來顯示使用者帳戶，圖中左方即爲樹狀顯示，而右方爲節點的資訊編輯畫面，其對於帳戶的編輯也提供較多的功能爲其優點。但使用介面過於複雜使用不夠直觀，樹狀顯示使帳戶資訊看

來過爲擁擠則爲其缺點。

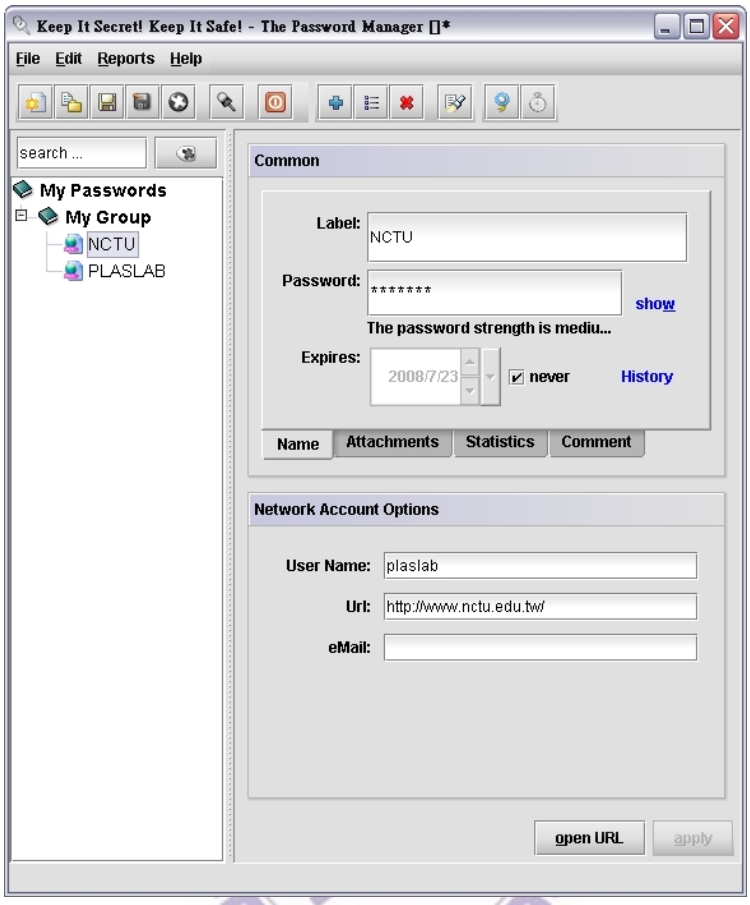


圖 2 - 3　KisKis

2.2.3　Pasword Safe

Password Safe，分別使用 C++ language 與 Java language 開發，而在 database 加密方面則使用了 Twofish encryption algorithm。

圖 2 - 4爲Password Safe的主畫面以及帳戶編輯畫面。與KisKis相同，使用tree方式作爲帳戶的分類管理；但與KisKis不同的是，Password Safe將編輯畫面獨立於一對話視窗中取代了將其置於畫面的右半邊，使得使用介面較爲清爽簡便。同時Password Safe提供了比前兩者更多的編輯功能像是password generator，產生亂數密碼取代使用者重複使用類似形式的帳戶密碼。Password Safe雖然將帳戶編輯畫面獨立於額外的對話視窗，然而將帳戶資訊置於樹狀分類中在帳戶數目過多時仍然造成畫面的擁擠與編輯上的某些不便。

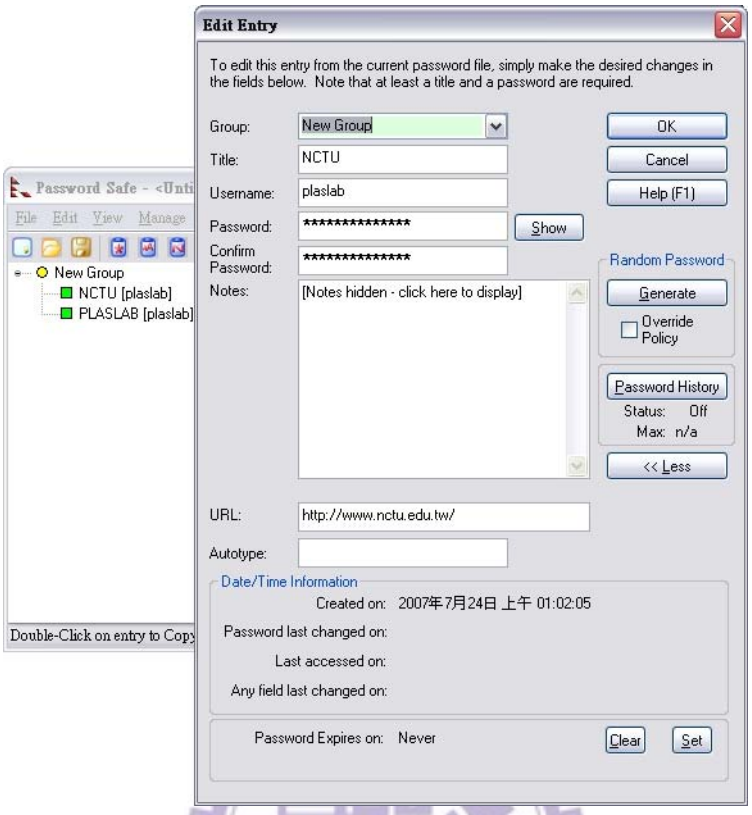


圖 2 - 4　　Password Safe

2.2.4　PassReminder

PassReminder 亦使用 Java language 寫成，然而他分別爲 Windows 與 Linux 開發了兩種不同的版本，Windows 版本又分爲需要 JVM 與不需 JVM 的兩種版本。檔案加密方面則使用了 Blowfish encryption algorithm。

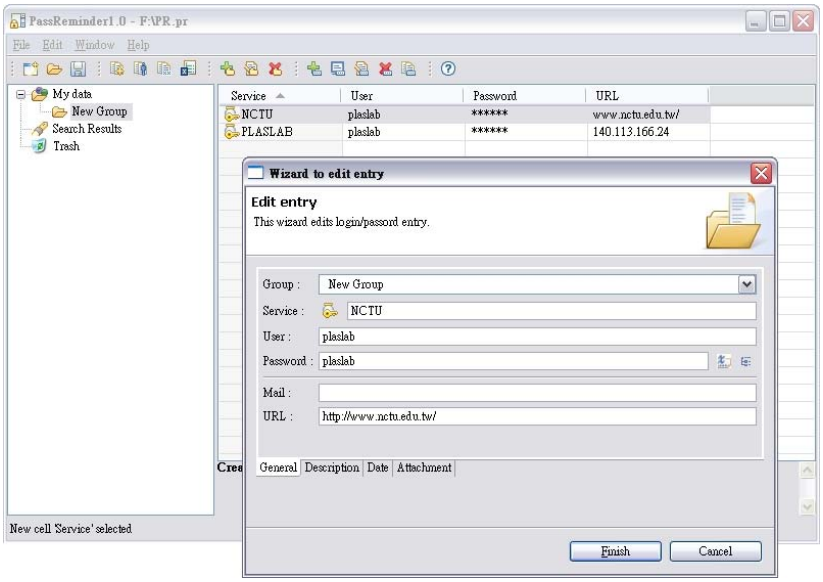


圖 2 - 5　　PassReminder

圖 2 - 5可以看到PassReminder的主畫面以及帳戶編輯畫面，其提供較爲完整的美工大大增加了系統的親和力，並且將帳戶資訊本身從帳戶分類的tree顯示，另外分開置於右半邊以table方式顯示，這樣的配置方式在使用上較爲便利與直觀。PassReminder在使用性大大優於前三者，且提供了語言的外掛機制能夠以plugin的方式加入不同的language外掛；但其缺點與UPM類似，對帳戶資訊只有簡單的記錄功能，帳戶的建立修改時間等一些實用的額外資訊卻沒有提供；功能過於陽春爲其美中不足的地方。

2.2.5 KeePass

KeePass，使用 C++ language 開發，目前只有 Windows 的.exe 安裝版本，然而亦有一名爲 KeePass X 的專案開發其分別於 Linux 與 MacOS X 的版本。Database 的加密方面提供 AES 與 Twofisn 兩種加密演算法供使用者選擇其一作檔案的加密。

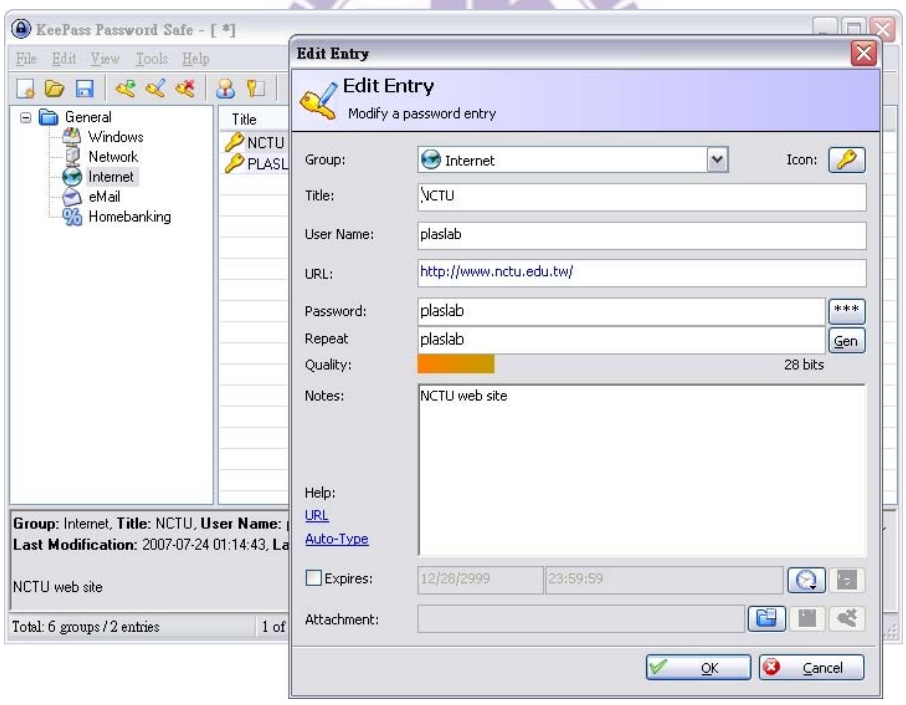


圖 2 - 6 KeePass

圖 2 - 6爲KeePass的主畫面與帳戶編輯畫面，與PassReminder有類似的美工GUI，然而提供了較PassReminder更爲完整的功能。缺點就是必須針對不同的OS作業環境找尋找不同的安裝版本。

第 3 章
系統設計

在系統進行實作前，本章對於應具有的功能以及相關檔案格式來做設定，並闡述本系統的組織架構。

3.1 功能需求分析

每一份的 User Account/Password pair 皆代表使用者在網路上的一筆個人資訊。除了對於存有多筆使用者帳戶資訊的 Database 有加密的必要之外，我們也要求對於 Database 中的每一份帳戶資訊都分別做加密的動作。

加密的部份爲了達到 Strong security 混合使用多種 Cipher。而 Cipher 以及其 Secret Key 的使用除了系統提供的一份預設值外，皆可由使用者自行設定調整。

混合使用 Cipher 以及自行設定 Secret Key 衍生出的問題就是 Key Management。Secret Key的新增、刪除以及更新都將由一Key-File來管理。以增加系統安全性爲由，Key-File提供匯出功能與Database存放於不同的地方。圖 3 - 1爲本系統的設計概念圖。

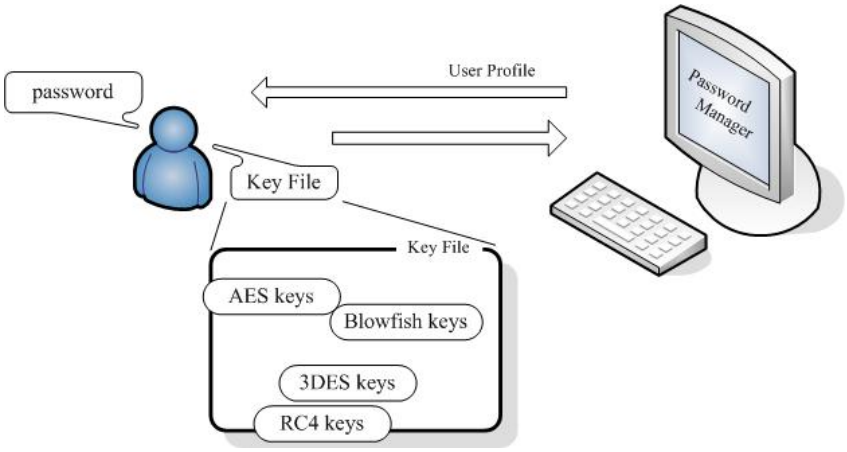


圖 3 - 1　系統概念

爲減輕使用者的負擔，上述功能都經過 GUI 包裝成直觀且容易使用的視窗介面；使用者僅需記得 User Password 以及持有用來解密的 Key-File 即可對於

Database 做編輯動作。除此，我們也需提供搜尋的功能方便使用者找尋某項 Database 中他所感興趣的資料。

3.2 系統架構概述

圖 3 - 2爲Personal Password Management System的架構圖。本系統大致可劃分爲三個部份：分別爲和User互動的GUI部份，Key Management以及存放使用者資料的Database。

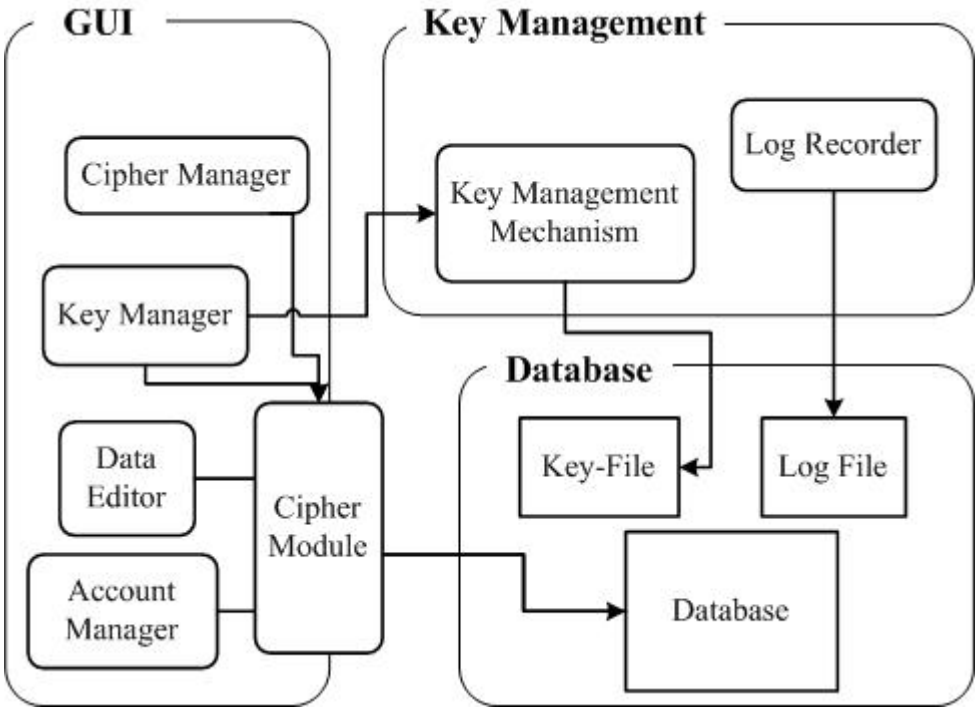


圖 3 - 2　系統架構

3.2.1 GUI

GUI 是銜接使用者及其帳戶資訊的接口，我們可以透過簡單的編輯方法來對資料做修改與更新。

- Cipher Manager：

  本系統混合使用多種 Cipher，使用者在編輯帳戶資料時可藉此設定欲使用的 Cipher Module，有提供 Default 值

- Key Manager：

  資料加密必須對 Cipher Module 設定對應的 Secret Key；除此，可以進行 Key Label 的新增、刪除以及更新某 Key Label 相對應的 Secret Key。

有提供 Default 值

- Cipher Module：

  使用的選取的 Cipher 以及其 Secret Key 設定，資料儲存時會經由此 Cipher Module 進行加密

- Data Editor：

  可在此編輯解密後的資料，儲存時再做加密的步驟

- Account Manager：

  可進行帳戶的新增、編輯與刪除等動作

3.2.2 Key Management

本系統最大的特點即是 Cipher 的混合使用，因此必須對複數的 Secret Keys 進行有效的管理。

- Key Management Mechanism：
  每一個 Cipher 擁有自己的 Key Array，記錄 Key Label 對應至那一把 Secret Key。加密/解密時查詢使用者設定的 Key Label，而後使用對應的 Secret Key 對 Cipher 進行初始化
- Log Recorder：
  使用者更新 Key 的最近幾筆設定會被記錄下來

3.2.3 Database

使用者資料以及 Key-File 經過加密後存放於磁碟中，基於安全考量，Key-File 保管於不同位置是較好的選擇。

- Database：
  包含使用者所有的帳戶資訊，以加密的形式儲存
- Key-File：
  包含 Cipher 與 Secret Key、Key Label、以及 Database 的加密設定
- Log File：
  存有由 Log Recorder 所記錄的最近幾筆 Key 變動資料

3.3 系統設計

在實作本系統前先針對我們要達到的目標及功能需求，對於檔案格式、使用者介面等等進行系統元件的相關設計。

3.3.1 Properties 格式設定與內容

Properties file的內容是本系統的一些參數設置，例如讀取使用者Database的File path、相對應的金鑰檔案Key-File path、以及系統預設使用的加密/解密演算法。Properties file以pwManager.property做爲其檔名，可透過GUI介面修改其內容，或是手動開啓予以編輯。以圖 3 - 3爲例：

```
1# pwManager.property
2#
3# Some properties of Password Manager
4#
5
6
7# filepath
8#
9# Default opened database path
10# If the filepath is empty or the database is not exists,
11# pwManager will open an default database
12#
13filepath=./src/database
14
15
16# keypath
17#
18# User's key-file path
19# If the keypath is empty or the key-file is not exists
```

圖 3 - 3　Property File 格式

每行以“#”字元開頭者表示註解，系統將予以忽略。參數的設定則爲“name=value”的形式，系統會將 value 值在讀入檔案後傳入 name 參數中。

3.3.2 Database (Data-File)格式設定與內容

Database是以Ciphertext形式儲存在磁碟中，但在讀取的過程中會先將Database暫時解密至副檔名爲“.tmp”的中間檔案。我們將.tmp中間檔設定爲圖 3 - 4的格式，使系統能夠依照此格式建構出原始的使用者資料。

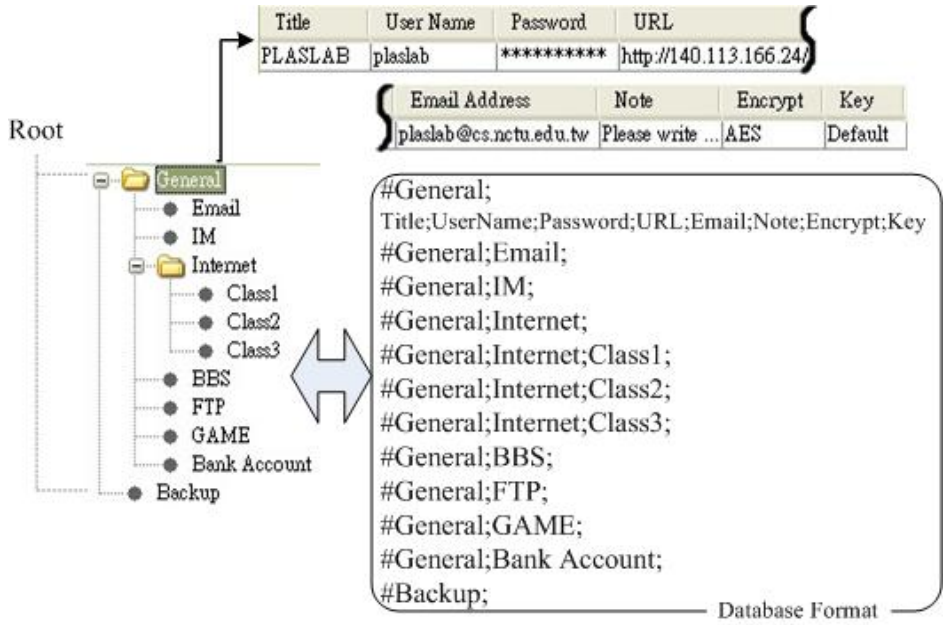


圖 3 - 4　　Database 格式

和 Properties file 不同的是，以“#”字元開頭的行在此處代表一個“Tree Node”，緊接在後沒有以“#”字元開頭的行則代表此 Node 中所含有的使用者帳戶；每一行又使用“;”字元來對欄位做切割。

以圖 3 - 4爲例，左方的Tree爲本系統在讀入右方的Database檔案後建構出的結果，而在第一個名稱爲“General”的Node中包含了一筆Title=PLASLAB，User Name=plaslab的使用者帳戶資料。此例中我們可以清楚看到“;”的切割作用。對Tree Node而言使用“;”做切割表示其“Parent/Child”的結構；而對使用者帳戶資料而言僅代表將其不同欄位做分隔。樹狀結構的Root根節點在系統中則是隱藏的狀態不加以顯示，也不能對其加以存取。

3.3.3　Key-File 設定

Key-File 在系統中也是以 Ciphertext 形式存在。Key-File 是將系統中對金鑰做管理的“Key Object”以 Object Serialization 的方式輸出至磁碟中並予以加密，其中包含以下資訊：

- 系統提供的 Cipher 名稱
- 加密的 Database 所使用的 Default Cipher
- 由 User Password 建構出的 Default Secret Key，以 Base64 型態呈現
- 用來解密 Database 的 Key Pair，以 Cipher - Secret Key 的形式存於 Array
- 每個 Cipher 各自的 Key Label - Secret Key HashMap，以 Vector 儲存

Key-Object的內容在4.2部份有詳細解說。

3.3.4 帳戶資訊欄位設定

本系統設計了圖 3 - 5中幾項較爲使用者感興趣的欄位，各欄位解說如下：

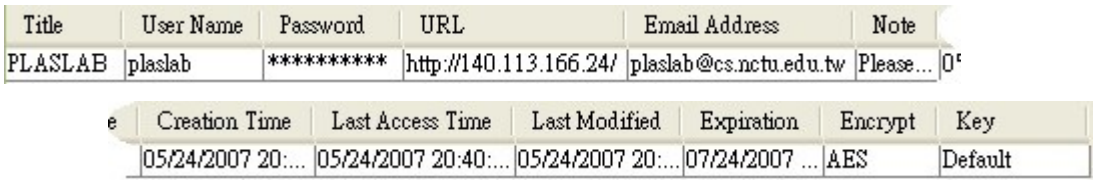


圖 3 - 5　欄位設定

- Title：
  給予此筆帳戶資訊一個具代表性的標題，可利用易於辨識的文字敘述幫助日後找尋資料使用
- User Name：
  所註冊的使用者名稱，即代表在此服務中用於識別使用者身份的依據
- Password：
  註冊登記的密碼，正確的密碼才能取得此帳戶存取的權限
- URL：
  網路服務的連結位址。若此服務並非網路服務，可記錄相關聯的網頁例如郵局帳戶的台灣郵政網站、ADSL 上網帳戶的 ISP 網站、或是某款遊戲的官方網站等
- Email Address：
  對於大部分的網路服務都必須使用 Email 來進行認證程序，此欄位提供使用者記錄其所用的 Email Address
- Note：
  此筆帳戶資訊的細節敘述、註解等雜項
- Creation Time、Last Access Time、Last Modified：
  這筆資料的建立時間、最後存取時間、以及最後一次修改時間
- Expiration：
  這筆資料的有效期限，可設定是否啓用此欄位
- Encrypt：
  使用何種 Cipher 加密/解密，預設值爲 AES
- Key：
  Cipher 在加密/解密時使用的 Secret Key，顯示的僅爲代表此把金鑰的 Key Label 值

在本系統開啓時預設顯示的欄位分別爲 Title、User Name、Password、URL、Email Address、Note、Encrypt、Key，其他建立、存取時間等欄位可自行設定是否顯示；Password 欄位可設定爲以“*”字元來保護使用者密碼或是用 Plaintext

顯示；URL 欄位依照使用者填入資料可選擇開啓相對應連結位置。

3.3.5 GUI 設計

本系統分別使用表格以及樹狀結構來讓資料的顯示符合使用者的閱讀習慣，如圖 3 - 6：

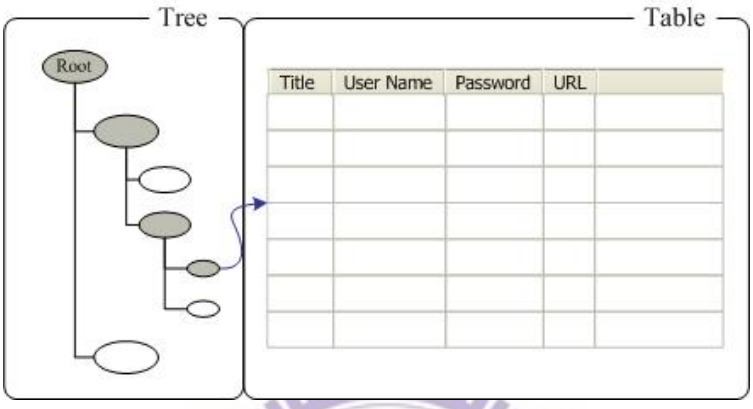


圖 3 - 6　GUI 設計

右方的表格中每列代表一筆帳戶資訊，使用者可將性質相似的帳戶記錄於同一份表格，表格內容存放在左方樹狀的節點中。使用樹狀結構的目的在於方便帳戶的分類整理，點選節點則會出現相對應的帳戶資訊表。

同樣的，對於資料的修改以及金鑰的管理我們也都設計相對應的圖形介面提供使用者能以簡單的步驟加以編輯。

第 4 章
系統實作

Triple DES 藉由 E-D-E 的步驟強化原本 DES 的加密功能。接下來各節將對 Cipher 的混合使用、Secret Keys 的管理、Database 的加密/解密流程等做詳細的說明與分析。

4.1 NodeObject 實作與管理

系統的核心是由使用者資料構成，其運作皆爲一連串對於存放資料的 NodeObjects 做新增/修改/刪除的動作，Database 亦爲 Nodes 依照使用者的 Key-File 設定經過加密儲存後的產物。我們以存放使用者資料的 NodeObject 作爲單位來建構本系統，並且使用樹狀結構來進行 Node 的管理。

4.1.1 NodeObject 結構

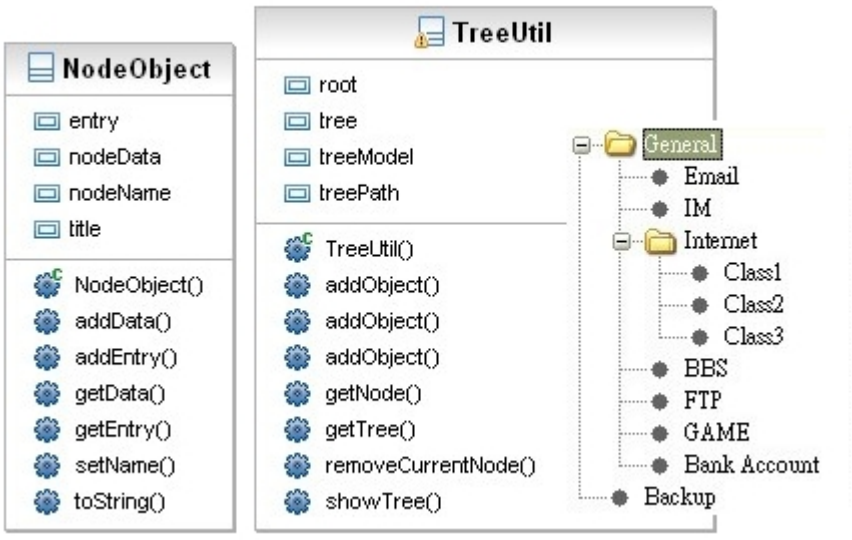


圖 4 - 1　NodeObject、TreeUtil 以及 Group Tree

如圖 4 - 1所示，由以下四個Field：title、entry、nodeData三個Vector型別的變數、與nodeName String型別的變數組成。title Field中存放3.3.4小節所設計的欄位名稱；entry Field使用Vector<Object>型別中存放使用者帳戶各欄位資料，

一個entry Vector代表一筆帳戶資訊；而nodeData Field所含的內容實質上就等於NodeObject主要內容，是由一筆或多筆entry變數，以Vector形式組成。複數的Nodes由樹狀結構組織起來顯示在系統的左分割視窗，Node內容則以表格形式顯示在系統的右半視窗。屆時name變數即爲此Node所顯示的名稱

4.1.2 NodeObject 管理

圖 4 - 1的右方是由NodeObject組成的樹狀圖，使用者可對其做新增/刪除/編輯Tree Node的動作。我們透過實作圖 4 - 1中的TreeUtil類別來達到以樹狀結構管理NodeObject的目的。TreeUtil提供addObject()、removeCurrentNode()等Method，讓使用者能以分組的方式管理不同類型帳戶資訊的能力。

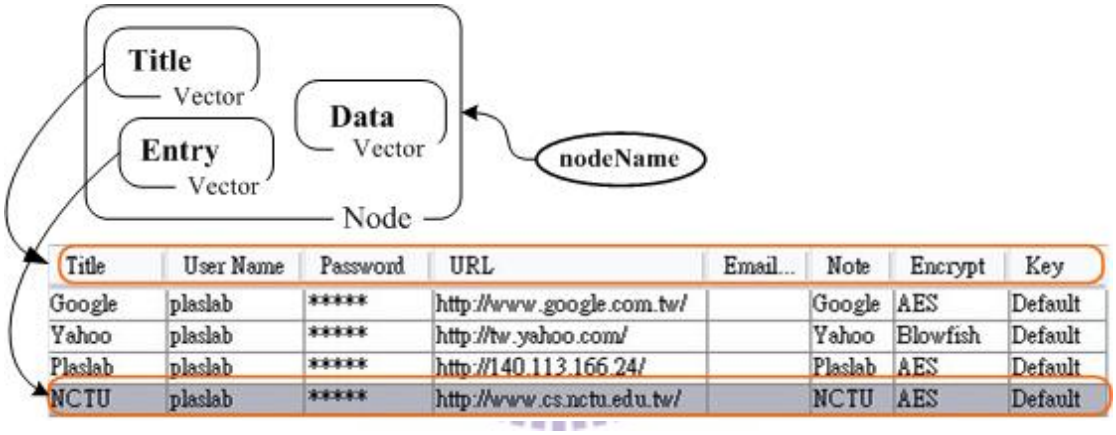


圖 4 - 2 NodeObject 管理

依照3.3.2小節所設定的Database檔案格式，我們能夠對讀出的資料進行切割並使用NodeObject::addEntry() Method依序將切割後的資料欄位加入entry變數中，形成一筆包含User Name/Password、加密Cipher、建立修改時間等資訊的使用者帳戶；再將一或多筆使用者帳戶透過NodeObject::addData() Method加入nodeData Vector。這時我們使用title Vector以及nodeData Vector建構出我們的使用者帳戶Table。如圖 4 - 2所示，此Table使用title變數建構Table Header，而Table顯示的內容即爲dataNode內的資訊，每一列代表一個entry Field。

使用 NodeObject 做資料的存取與 TreeUtil 來做 Node 的管理是系統最基本的運作架構。

4.2 Key-File 實作

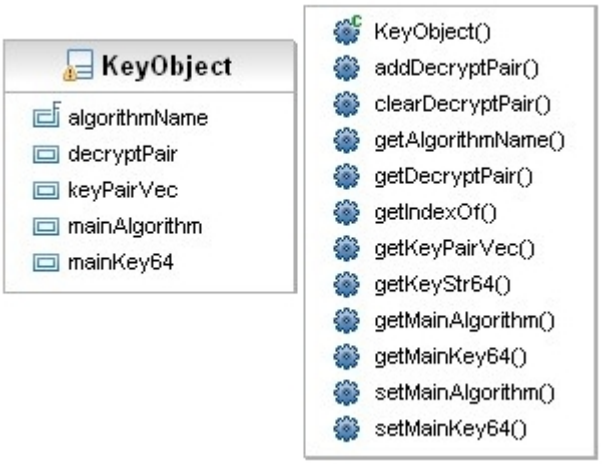


圖 4 - 3 KeyObject

混合使用Cipher和提供使用者自行設定金鑰功能的關鍵就在於Key-File，Key-File是KeyObject經過加密之後得到的檔案。圖 4 - 3爲KeyObject的實作內容，先對於KeyObject中各Field做說明：algorithmName Field定義了五種由系統提供的Cipher，使用者能夠對於選定的欄位任意更換使用由系統提供的Cipher來做加密；mainAlgorithm和mainKey64 Field分別記錄解開database，由使用者設定的Default Cipher與對應的Default Secret Key，使用這兩個變數的原因在4.5.1小節會做詳細說明；decryptPair Field爲自訂型別PairUtil的變數，存放檔案各Entry加密使用的Cipher-SecretKey pair，database解密時必須透過decryptPair才有辦法將檔案資訊還原；keyPairVec Field是型別爲Vector< HashMap<String,String> >的變數，每個Cipher都有自己對應的HashMap<String,String>，內容是使用者自行設定的KeyLabel-SecretKey資訊，keyPairVec將各Cipher的HashMap置於一Vector中加以控管。

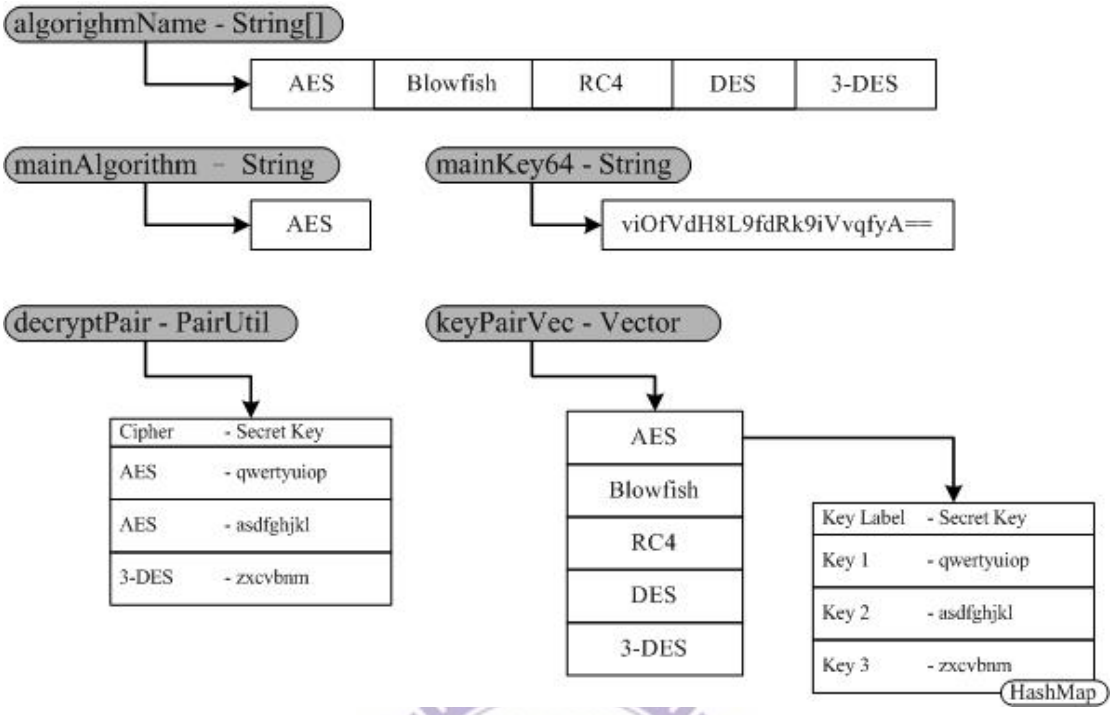


圖 4 - 4　KeyObject 實例

KeyObject在系統運作時的結構如圖 4 - 4所示，分別記錄了由系統提供的Cipher、相對database使用的主要加密演算法與金鑰、用來逐列解密的Cipher-SecretKey pair以及所有Cipher都各自擁有一份的Secret Key HashMap。

Secret Key HashMap 的用意在於提供使用者能夠自行對資料欄位做加密設定；將使用者設定的 KeyLabel-SecretKey Mappint 狀態存入。在對使用者帳戶做逐列加密動作時我們將每一步驟使用的 Cipher 和對應的 Secret Key 寫入 decryptPair Field 中，其 PairUtil 的型別存放每一步驟使用的 Cipher-SecretKey pair。

另外一題的是，我們在各 Field 中保存的 Secret Key 值並非 Cipher 直接使用的 Byte 陣列型態的 Secret Key，而是取其編碼值後做 Base64 轉換爲 String 型別存入；mainKey64 Field 中的 Key String 則是透過 User Password 經過 Hashing 的值，再轉換爲 String 所得到。

藉由 KeyObject 的實作我們能夠對不同的資料做不同層次的加密保護，即使不做設定也提供一份了 Default 設定值。

4.3 金鑰管理 Key Management

Data Security 也意味著 Keys Security，使用 Model-View-Controller (MVC) model 實作 Key management 提使用者能以簡易的動作對金鑰做有效管理。

4.3.1 金鑰管理 MVC model

圖 4 - 5是系統實作出的Key Management MVC model：

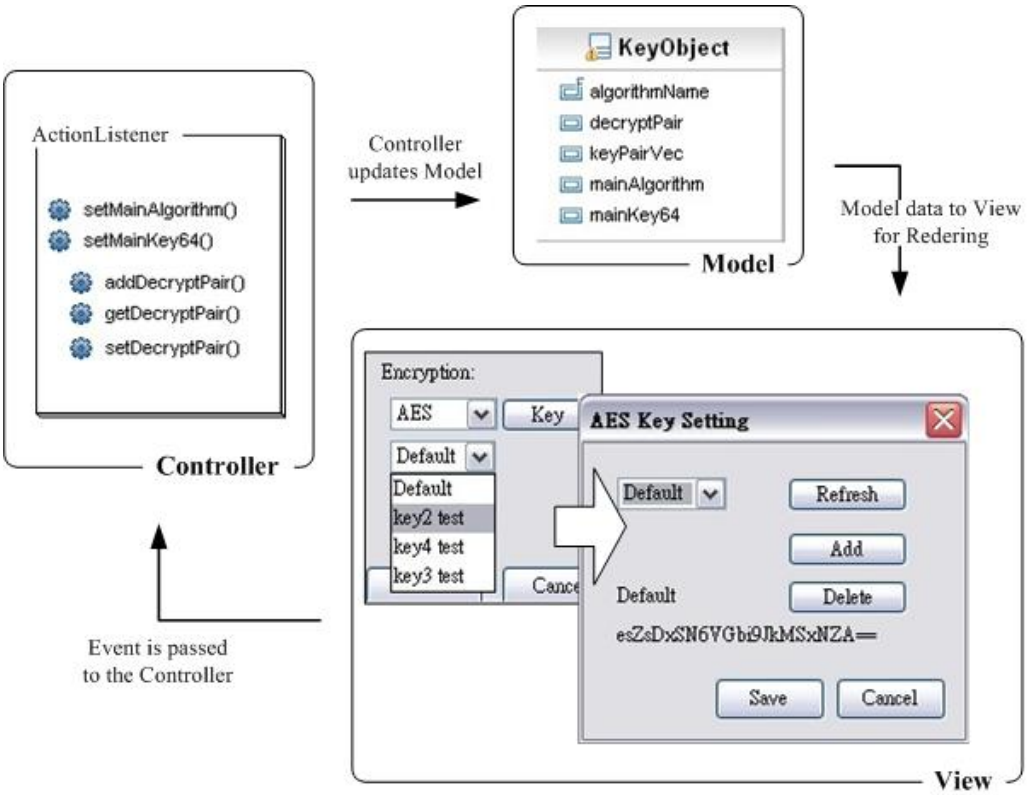


圖 4 - 5　金鑰管理 MVC model

- Model
  以KeyObject的Field變數 (4.2小節) 組成，存放系統當下Cipher與Secret Keys的設定。除了algorithmName Field爲一固定值外，使用者透過GUI介面做設定時內容將會更新，而decryptPair只在檔案儲存時記錄每個欄位的加密設定
- View
  將 Model 內容以 GUI 顯示。View 會根據 Model 的狀態變更目前顯示，若有新增或移除等資料修改，Model 元件在內容更新後會重新繪製 View 元件
- Controller
  編輯資料時會觸發事件由 Controller 進行 Model 元件的更新。透過 GUI

做資料的修改系統會發出 ActionEvent 的事件，Controller 元件接收此事件後更新 Model 中相對應的欄位

使用MVC架構能夠降低對金鑰管理的負擔，使用者選定加密目標欄位後可對於Cipher與Secret Key進行設定。如圖 4 - 5的View元件中的視窗所示，使用下拉式選單選定要使用的Cipher後，可在其下方的另一下拉式選單選擇要使用的Key。若要進行Key的新增/刪除/更新，僅需點選“Key” Button後再做相對的設定即可。GUI的顯示由Model元件目前內容所決定，若有變更會即時更新顯示。

4.3.2 金鑰設定

選擇 Key 的下拉式選單中顯示的內容僅爲作爲代表的 Key Label，在檔案儲存時系統會在 KeyObject 的 keyPairVec Field 中以 getDecryptPair() Method 找出相對應的 Secret Key。Key 的新增可以設定喜好的 Key Label 名稱，Controller 元件在接收新增 Key 的事件後會對此 Key Label 自動 Generate 一個對應於選擇 Cipher 的 Secret Key，再將此 KeyLabel-SecretKey pair 以 addDecryptPair() Method 新增至 Model 中。相同的，Key 的 Refresh 也是系統在接到更新 Key 事件後，對於選擇的 Key Label 重新 Generate 一個新的 Secret Key，並且呼叫 setDecryptPair() Method 做資料更新。要注意的是，系統對各 Cipher 提供一 Default Key Label，對應的 Secret Key 則是 User Password 透過 MD5 Hash Function 產生的 Byte Array 所建立；此 Default Key Label 無法進行更新，只有在變更 User Password 或是 Default Cipher 時由系統自動更新。

4.3.3 加密設定記錄

decryptPair Field以PairUtil的自訂型態記錄資料欄位加密/解密的設定。如圖 4 - 6，PairUtil是以ArrayList< String[] >型態的pairArr Field實作。

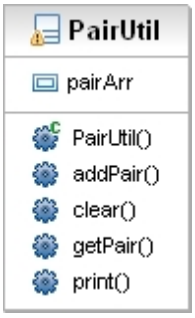


圖 4 - 6 PairUtil

儲存檔案將加密使用的 Cipher-SecretKey 儲存至二維 String Array 後依序加入 ArrayList。讀取檔案則反向操作依序取出 pairArr Field 中的 Cipher-SecretKey pair 設定，並依序解密資料欄位。

4.3.4 金鑰匯出

KeyObject 藉由 implements Serializable Interface，先以 ObjectOutputStream 輸出爲暫存檔形式儲存至磁碟，再依照 mainAlgorighm 與 mainKey64 Field 的設定將暫存檔加密成爲 Key-File。反的來說，使用 Key-File 解密 Database 時，需要先選擇系統解密使用的 Cipher 以及輸入 User Password，若正確無誤則可解開 Key-File 檔案成爲暫存檔形式，再藉由 ObjectInputStream 重新建構成 KeyObject。Database 的讀取需要取得 KeyObject 的 decryptPair Field 做欄位的解密。

Title	User Name	Password	URL	Email ...	Note	Encrypt	Key
Google	plaslab	*****	http://www.google.com.tw/		Google	AES	Default
Yahoo	plaslab	*****	http://tw.yahoo.com/		Yahoo	Blowfish	Default
Plaslab	plaslab	*****	http://140.113.166.24/		Plaslab	AES	Default
NCTU	plaslab	*****	http://www.cs.nctu.edu.tw/		NCTU	AES	Default

Title	User Name	Password	URL	Email ...	Note	Encrypt	Key
Google	plaslab	*****	http://www.google.com.tw/		Google	AES	Default
Plaslab	plaslab	*****	http://140.113.166.24/		Plaslab	AES	Default
NCTU	plaslab	*****	http://www.cs.nctu.edu.tw/		NCTU	AES	Default

圖 4 - 7 Key Export 結果

Key Export的機制爲，能夠藉由選擇Cipher和Key Label來決定Export的Key-File具有哪些欄位的保密功能。圖 4 - 7的例子爲我們對於Cipher Blowfish且Key值爲Default時，希望此欄位受到保護，而輸出的結果也如圖所示，第二列爲不顯示內容的資料。

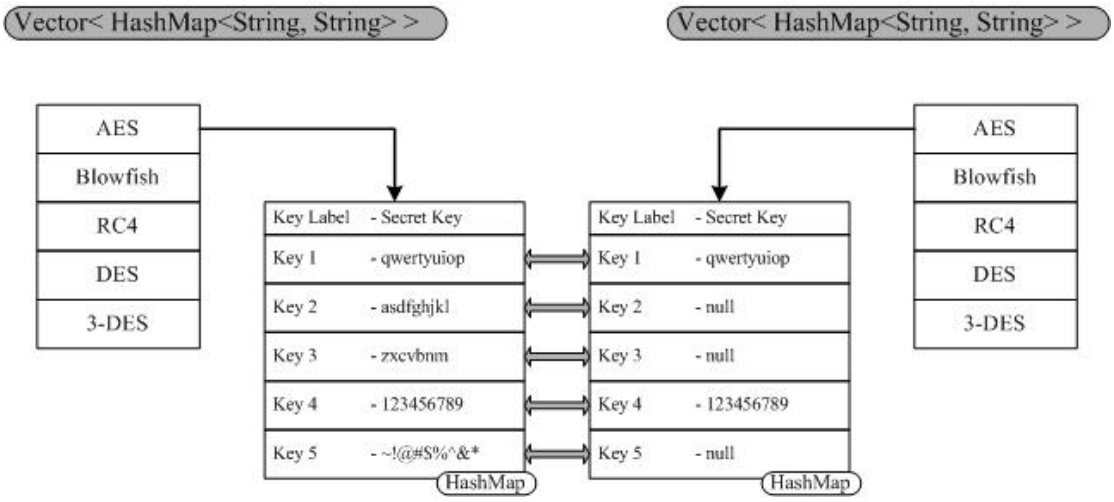


圖 4 - 8　　Key Export 機制

如圖 4 - 8，實作此機制的方法是，額外使用跟KeyObject keyPairVec Field擁有相同結構的Vector< HashMap<String, Boolean> >，把HashMap<String, String>原本的KeyLabel-SecretKey pair複製一份爲KeyLabel-isNull pair形式儲存。isNull的Default值也就是原本的Secret Key，不做任何設定Export的Key-File即可解開原本Database所有資料欄位；若有隱藏欄位的設定，isNull欄位將會塞入null。系統在匯出decryptPair時向此複製的KeyLabel-isNull pair詢問Cipher與Secret Key，並且將值寫入。而下次使用此設定匯出的金鑰讀取Database時，會忽略所有decryptPair內金鑰欄位值爲null的帳戶資訊，便完成了保護欄位的目的。

4.4　帳戶管理 Account Management

4.4.1　帳戶管理 MVC model

帳戶管理亦使用MVC model實作，拆爲Tree與Table兩部份，分別顯示對帳戶設定的分類與帳戶的詳細資訊。圖 4 - 9是系統實作的MVC model：

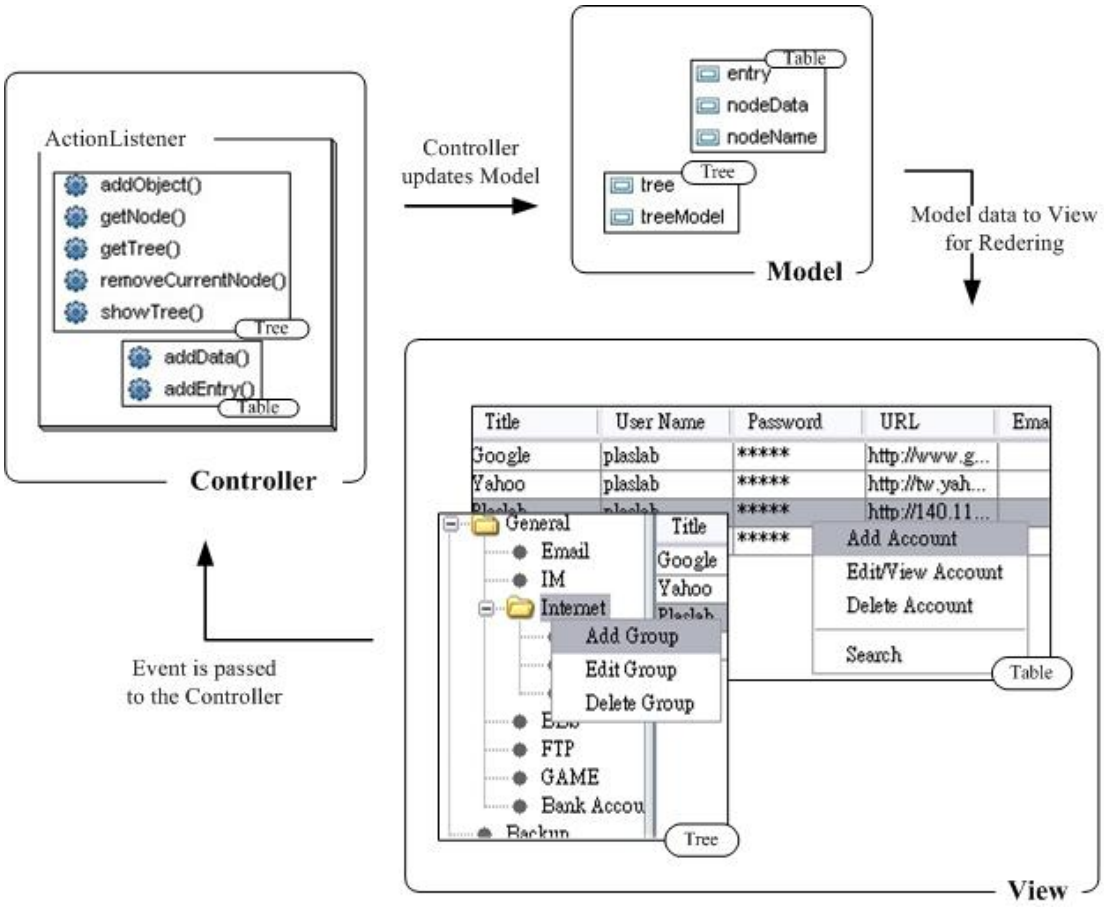


圖 4 - 9　帳戶管理 MVC model

- Model

  使用TreeUtil與NodeObject (4.1小節) 的Field變數構成，使得此Model有Data Entry→Data Node→Tree的形式。每筆帳戶資訊對應於NodeObject的entry Field (也就是Table的一列)，群組資訊由TreeUtil類別的tree、treeModel接收結構上與狀態上的更新，其中tree實際上是由一個或多個nodeData組成

- View

  tree Field 使用以 treeModel Field 爲參數的 Constructor 建立並且顯示 Tree GUI，內容爲 tree Field 的即時資訊；Table GUI 依選取的 TreeNode 不同有不同的顯示值，顯示內容爲選取 NodeObject 的 nodeData Field

- Controller

  分別使用 Tree 的 TreeSelectionListener 與 Table 的 ListSelectionListener 監聽使用者對 GUI 的選取狀態，若有資料設定上的變動系統會抓取新的輸入值，分別對 Tree Model 與 Table Model 內容更新，並且重新繪製 GU 元件

4.4.2 帳戶管理

藉由MVC model的實作可以輕鬆達到帳戶管理的目的，選擇MenuBar的Edit選項或是對選定的Table Entry以滑鼠右鍵開啓帳戶修改的對話窗做帳戶的編輯。圖 4 - 10是新增一筆使用者帳戶的例子：

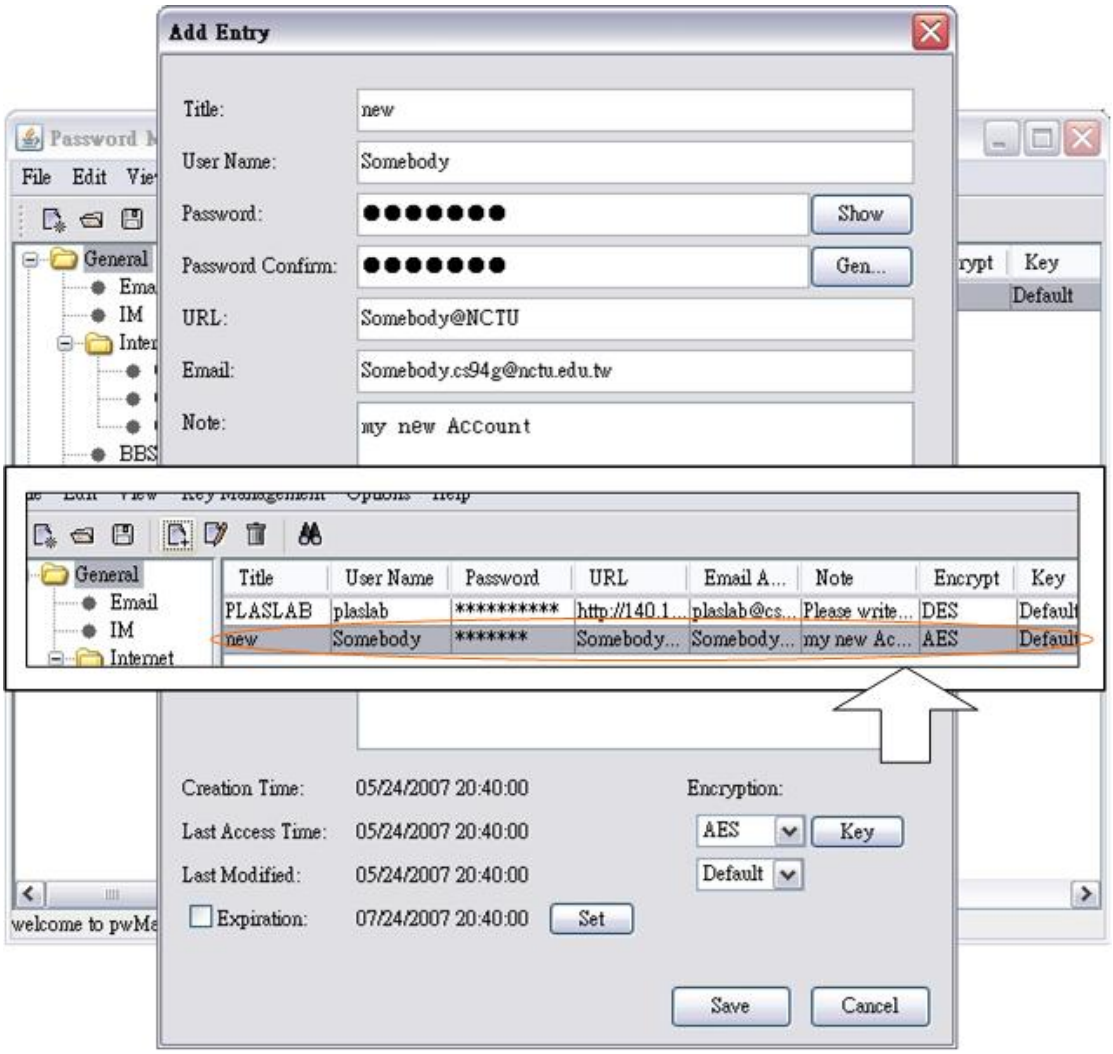


圖 4 - 10　帳戶管理範例

在跳出的對話窗內可以進行帳戶資料的設定。Password欄位Default值是不顯示內容的，但可點選“Show” Button改以Palintext顯示；其下方的“Gen…” Button提供亂數產生User Password的功能。對話窗下方是關於此帳戶的產生、修改時間等資訊，“Expiration”欄位可選擇是否啓用，若此筆帳戶達到設定的保存期限後將以不同顏色來顯示此帳戶。右下方的“Encryption”用以設定此筆帳戶的加密設定，同於4.3.1小節的說明。完成設定後會新增一列使用者帳戶接續在

Table中。

4.4.3 群組管理

圖 4 - 11即爲新增一個根節點群組的範例，對於樹狀結構可進行群組節點的新增/修改/刪除等動作：

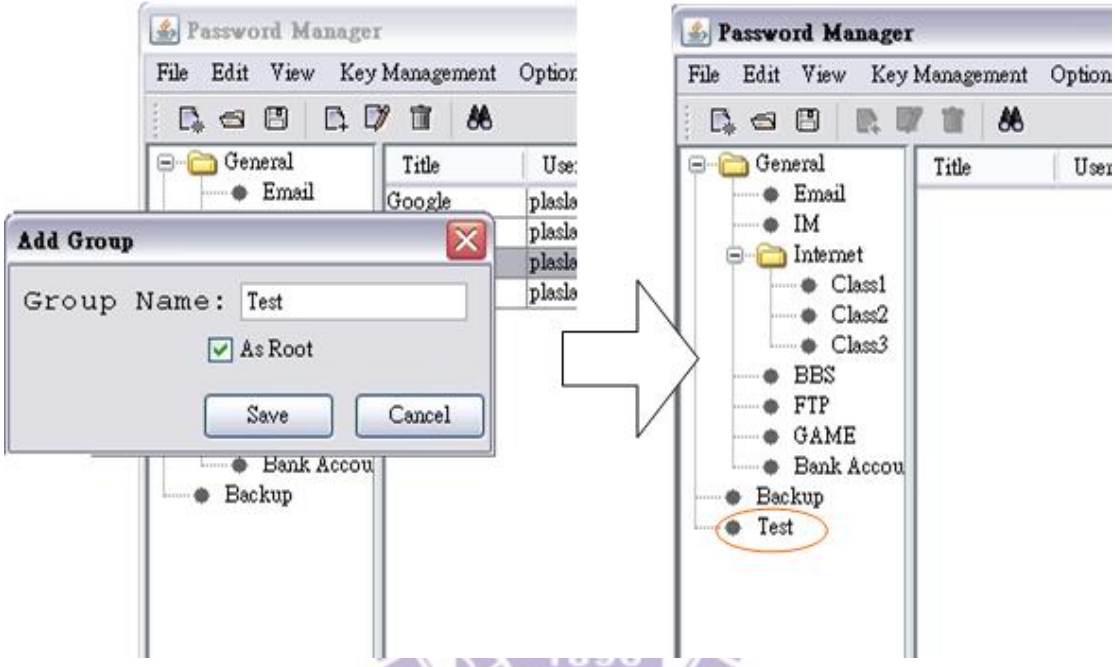


圖 4 - 11　群組管理

節點的修改也如圖所示的跳出 Edit Group 對話窗，不同的是沒有 As Root 的選項，且只能對名稱做更改；而刪除節點則需要注意，刪除對象的所有子節點會一併被刪除。

4.5 Database 加密/解密流程

Database 的加密過程使用了多種 Cipher，此節說明如使用 Key-File 對於 Database 加密/解密的流程。

4.5.1 Key-Object 設置

在對資料做加密前首先釐清 Key-Object 各 Field 如何產生，並且在加密時是如何作用。

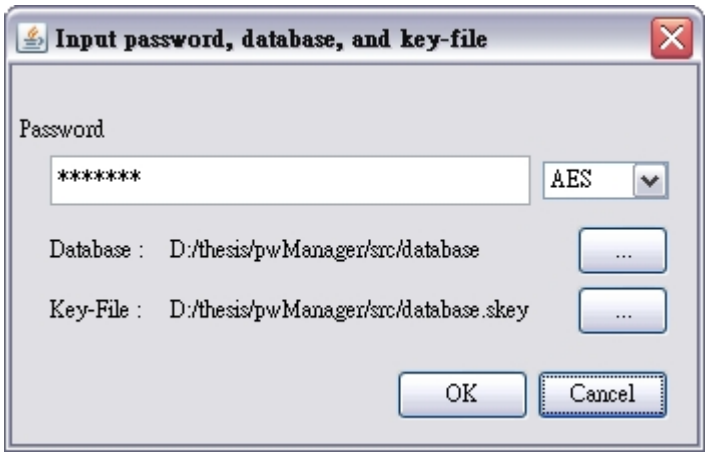


圖 4 - 12　Startup dialog

系統初在次執行時 (新增 Database)，會要求使用者設定 Default Cipher 與 User Password，並且呼叫 KeyObject 的 Constructor 將使用者設定作爲參數傳入建構 KeyObject，分別把值存入 mainAlgorithm Field 與 mainKey64 Field，同時生成只有空殼的 Default Database。KeyObject 只在首次產生 Database 時做建構，此後都僅對其做修改的動作。

圖 4 - 12是開啓Database的Startup Dialog畫面，若是新增Database的情況，使用者僅需對Password與Cipher欄位做設定，Database與Key-File欄位將會爲空，在儲存之後才會將儲存路徑寫入pwManager.property (3.3.1小節) 檔案，往後每次執行時會讀取pwManager.property最後一次的設定當作Default的開啓路徑。

KeyObject在尚未對資料做任何更新情況下，decryptPair Field內容爲空，而keyPairVec僅存在Default Key Label對應至mainKey64 Field值的一筆資料，此Default值是由User Password進行MD5 或是SHA-256 Hashing後，再將得到的byte[]做Base64 轉換爲String型別而得。對於各Cipher新增Key Label，系統會對於此Key Label Generate對應的Secret Key，並且以HashMap型別加入keyPairVec中。Key Label的修改/刪除則比較麻煩，除了更改HashMap內容外，我們需要對樹狀的使用者資料TreeUtil (4.1.2小節) 做Depth First Search (DFS)，將使用此Key Label的資料重新設定爲Default值。

decryptPair Field僅會在資料儲存時做更新，對TreeUtil做DFS且進行資料的逐筆儲存，成爲3.3.2小節設定的檔案格式。其過程中依序將每一資料欄位的加密設定以Cipher-SecretKey pair的形式加入decryptPair Field；若不是資料欄位而是TreeNode，我們則以defaultKey64 作爲其Secret Key的值。

圖 4 - 13對KeyObject的設置作整體的View：

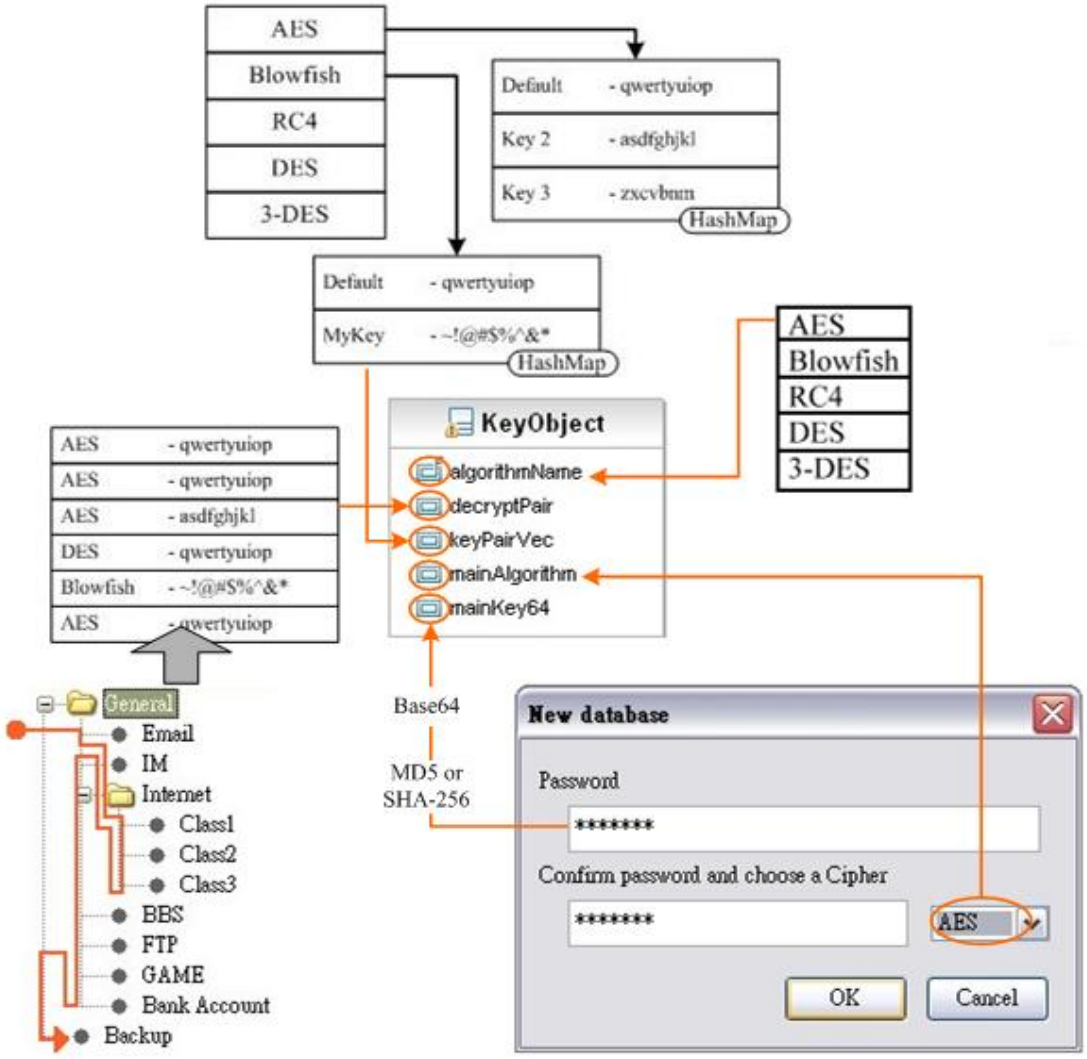


圖 4 - 13　KeyObject 設置

4.5.2　Database 加密

加密存檔可細拆爲幾個子步驟，如圖 4 - 14所示：首先對於資料做Base64轉換以3.3.2小節的形式做欄位切割；接下來以行爲單位依照使用者設定做欄位加密輸出至.tmp中間檔，同時將加密資訊存入KeyObject的decryptPair Field；最後一個步驟則是將中間檔以使用者設定的Default Cipher與Default Key (也就是mainAlgorithm Field與mainKey64 Field) 加密爲Database存入磁碟中，並刪除前一步驟產生的.tmp中間檔。

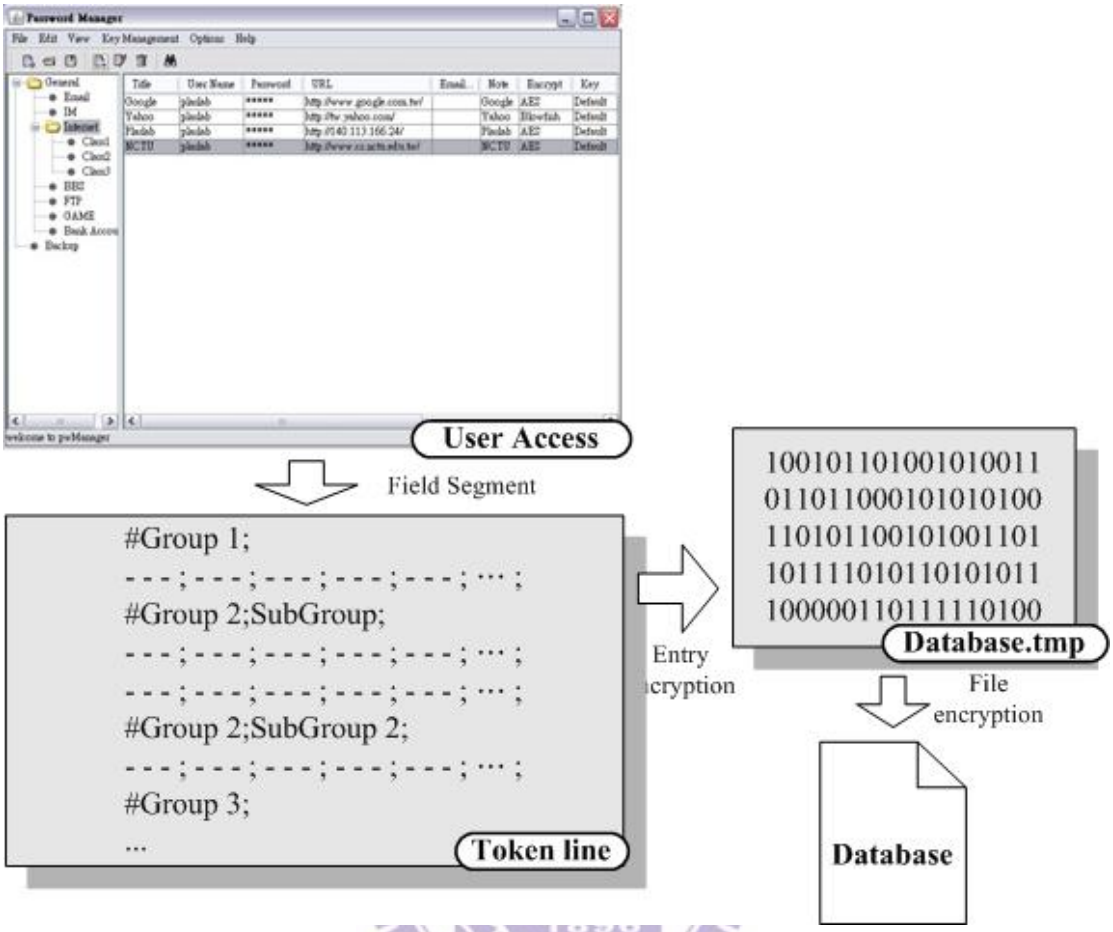


圖 4 - 14　Database 加密

Key-File 在每次 Database 存檔都會做同步更新，KeyObject 以 Object Serialization的方式輸出至磁碟中存爲中間檔，皆者與Database加密的最後一步相同使用Default Cipher與Default Key對中間檔加密成爲Key-File (圖 4 - 15)。這裡有一點要加以說明的是，Key Expotr的動作雖然也是將KeyObject以相同的方法加密爲Key-File，然而其decryptPair Field的欄位資訊會依照金鑰匯出時的設定以Null值填入相對應的設定欄位。

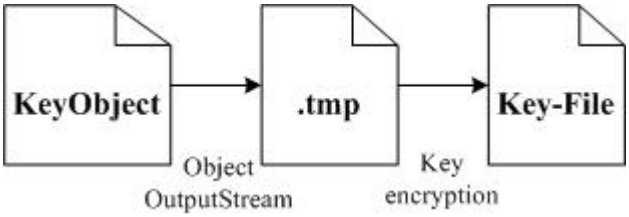


圖 4 - 15　Key-File 加密

4.5.3 Database 解密

將Database解密的步驟就是依照加密的步驟以反方向操作。如圖 4 - 16所示，使用者開啓圖 4 - 12的Startup Dialog後輸入正確的密碼，並且選擇Database與Key-File使用的Default Cipher，系統依照輸入設定將Database解密爲.tmp中間檔，同時把解開的Key-File讀入成爲KeyObject。此時如果發生錯誤會發出警示訊息通知使用者確認輸入是否正確；否則系統以行爲單位，依照Key-Object的decryptPair Field資訊解密讀入行。解密的行是Base64 的形式且以3.3.2小節設定的形式切割欄位，在藉由欄位分析恢復爲原始資料，以Tree與Table的形式呈現。

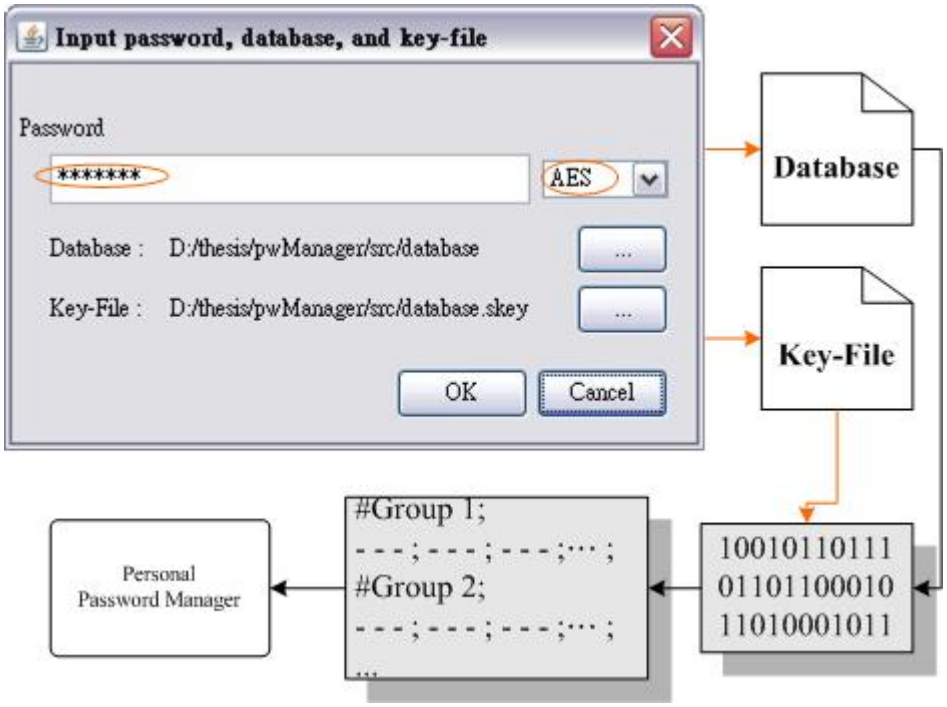


圖 4 - 16 Database 解密

使用選擇性 Export 的金鑰解密檔案則爲特例，在擷取 decryptPair Field 時被保護的欄位資訊將會是 Null 值，因此系統在判別讀入的 decryptPair Field 相對欄位爲 Null 時，此筆資料欄位將予以隱藏。

4.6 開發環境

與一般的 Web Services 相同，Platform Independent 爲其特色。網路的使用不分平台，本系統提供使用者管理個人的網路服務帳戶資訊也以此爲前提進行設計。本系統建構在 Java SE 6 環境下，利用 Java 的跨平台特性來達到系統

Portable 的需求。以下使用兩種不同開發環境，分別在 Windows 與 Linux 下對本系統的可攜性做測試；另外使用隨身碟測試使用者的 Data File 以及 Key File 在不同環境下是否依然可以如常運作，以及使用者匯出金鑰隨身攜帶的功能。

Desktop computer - window®

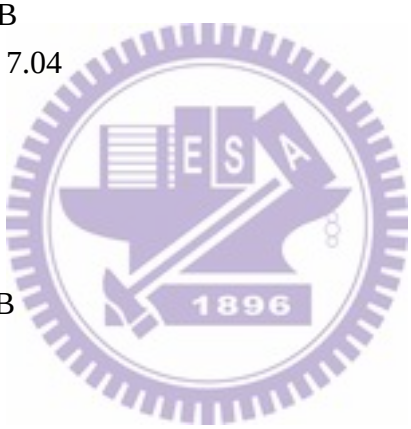
- CPU: AMD Athlon™ 64 Processor 3000+ 1.8 GHz
- RAM: 1.00 GB
- OS: Windows XP Professional
- JDK: 6.0
- JRE: 1.6.0

Desktop computer - linux

- CPU: Intel® Celeron® CPU 1.70 GHz
- RAM: 1.00 GB
- OS: ubuntu 7.04
- JDK: 6.0
- JRE: 1.6.0

Thumb Drive

- Flash: 128 MB



第 5 章
系統測試

本章針對本系統的混和加密以及金鑰的匯出進行測試。分別對於使用多種 Cipher 與設定多把 Secret Keys 來測試系統的正確性，以及使用對於特定欄位進行保密設定的金鑰來測試解開 Database 後目標欄位的保密性。

5.1 系統概述

進行加密與金鑰匯出測試前，以新增一個帳戶群組的範例，對系統做概略的介紹。圖 5 - 1是開啓新的Database時要求使用者輸入密碼，以及使用的Default Cipher。

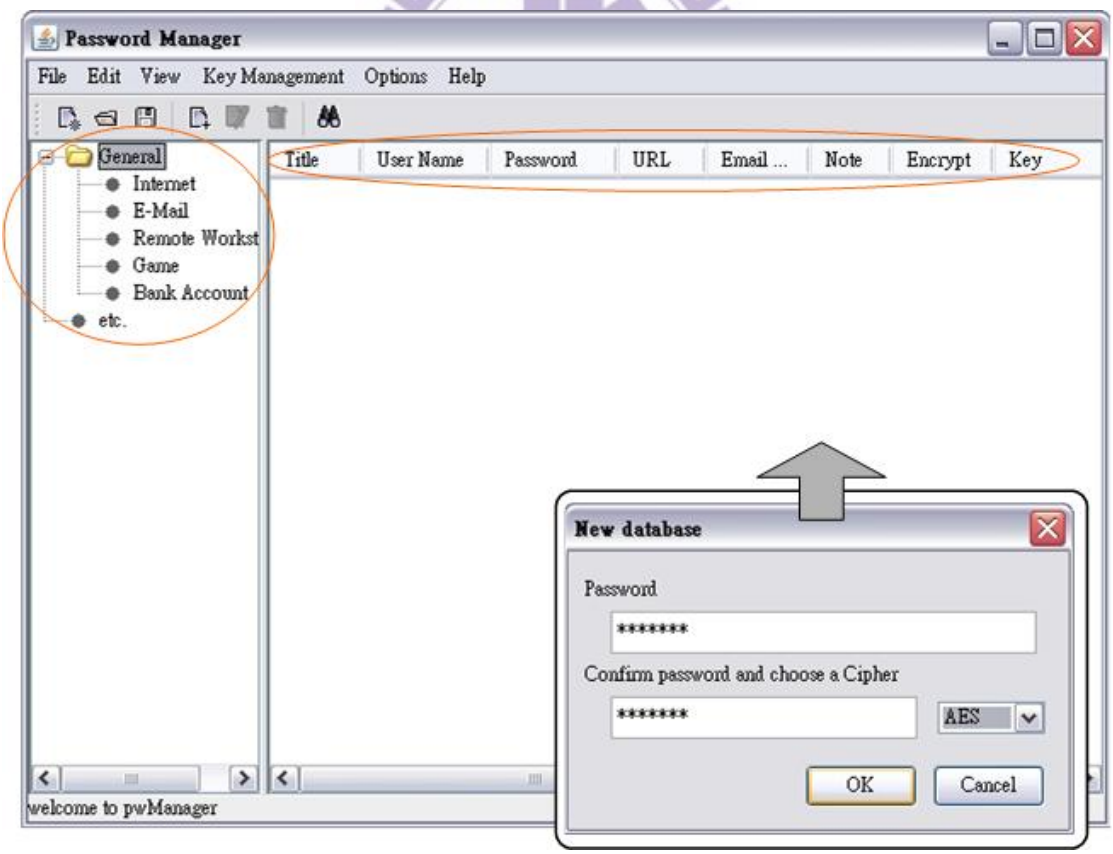


圖 5 - 1　新增 Database

設定完成後系統會開啓Defaule Database。左分割視窗是對於帳戶的群組分類樹，點選節點後會在右分割視窗顯示此節點中存放的使用者帳戶資料。可藉

由上方的MenuBar或是直接在要更改的欄位上點選滑鼠右鍵即可進行編輯。File Menu可以選擇開啓現有的Database，若選擇開啓舊檔則會跳出圖 4 - 12的確認視窗；Edit Menu是對檔案編輯的選項；另外Key Management則提供使用者金鑰的設置以及匯出的功能。

接著我們在Internet的節點下新增一個名爲Portal的新群組，並且在此群組中加入四筆資料，其結果如圖 5 - 2所示：

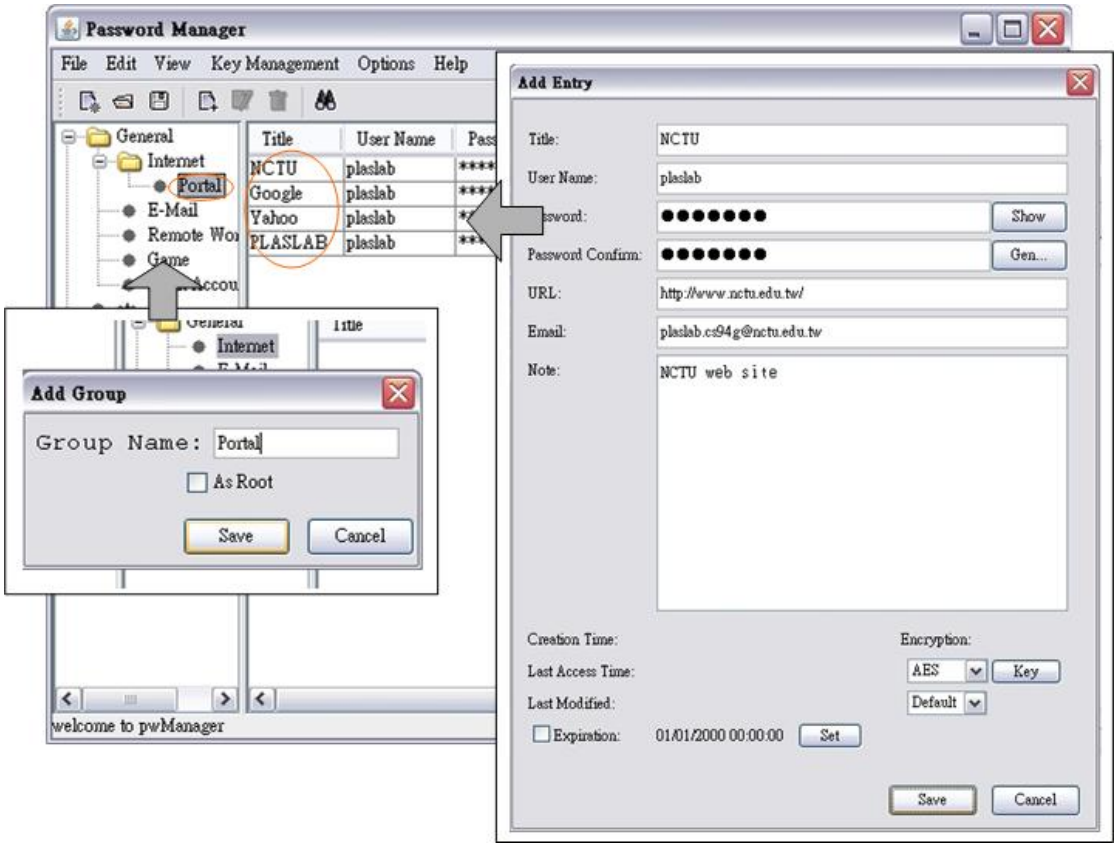


圖 5 - 2　群組與帳戶編輯

節點的新增我們可以選擇是否要以跟節點的模式加入樹中，若As Root方塊沒有勾選則新節點會以成爲目前選擇的子節點。帳戶的新增我們可以在各對應欄位輸入資料，Show Button可將Password欄位以明文顯示。Gen… Button提供使用者一亂數產生密碼的功能，更正密碼形式過度重複的缺點。左下則是此筆資料的建立更新時間等，Expiration核取方塊可選擇是否啓用密碼有效期限的通知，並且用Set Button來做日期設定。右下方的Encryption是此筆資料的加密設定，簡化繁複的金鑰管理功能。最後我們得到如圖 5 - 3的Table顯示各使用者設定的帳戶資訊。

Title	User Name	Password	URL	Email...	Note	Encrypt	Key
NCTU	plaslab	*******	http://www.nctu.e...		NCTU web site	AES	Default
Google	plaslab	*******	http://www.googl...		Google	AES	Default
Yahoo	plaslab	*******	http://tw.yahoo.co...		Yahoo!	AES	Default
PLASLAB	plaslab	*******	140.113.166.24		PLASLAB	AES	Default

圖 5 - 3　　帳戶新增成功

5.2　　檔案加密測試

接續5.1小節的例子，我們將新增的四筆帳戶資料分別設定爲不同的Cipher以及Secret Key來測試我們對於不同欄位的加密與解密是否能夠正確運行。

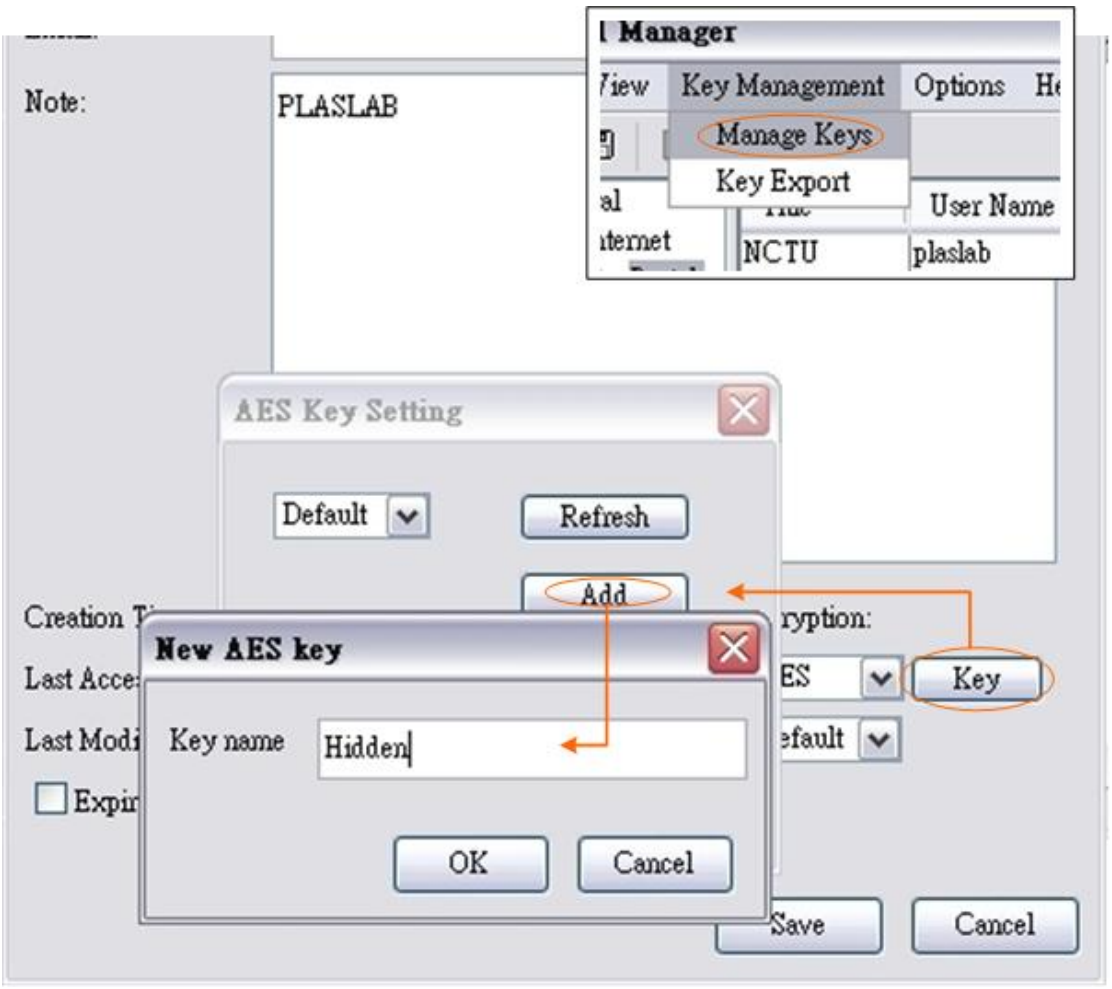


圖 5 - 4　　金鑰設置

可以使用MenuBar的KeyManagement選單來對金鑰做設定，這裡透過直接編輯帳戶的方式分別更改各欄位的加密設定。開啓編輯帳戶對話框後點選Key Button，因爲PLASLAB欄位目前使用AES Cipher，故跳出AES Key Setting Dialog；若要對其他Cipher做設定只要先選擇Cipher，再點選Key Button即可。

目前選項只有Default一項，選擇Add來新增“Hidden”的Key Label後，可以發現多了Hidden的選項，此時系統對於Hidden Label產生一對應的Secret Key (圖 5 - 5) 並且push入4.2小節說明的keyPairVec Field中。下方的亂碼字串表示Default Key Label所對應以Base64 形式表現的Secret Key值。重複上述步驟，將Blowfish Cipher同樣新增一名爲Hidden的Key Label。

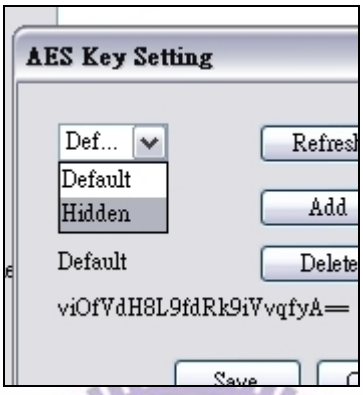


圖 5 - 5　　Key 設置成功

接下來分別設定NCTU與Google的欄位使用AES、Yahoo欄位使用Triple-DES、而PLASLAB欄位使用Blowfish來加密，並且把NCTU與PLASLAB的Key欄位設定爲剛才加入的Hidden Label，圖 5 - 6爲修改後的Table：

Title	User Name	Password	URL	Email...	Note	Encrypt	Key
NCTU	plaslab	*******	http://www.nctu...		NCTU web site	AES	Hidden
Google	plaslab	*******	http://www.goo...		Google	AES	Default
Yahoo	plaslab	*******	http://tw.yahoo....		Yahoo!	DESede	Default
PLASLAB	plaslab	*******	140.113.166.24		PLASLAB	Blowfish	Hidden

圖 5 - 6　　金鑰設置結果

設定結束後儲存檔案，我們來檢視加密的結果。圖 5 - 7裡我們分別可看到3.3.1小節設定的properties檔案pwManager.property、database、以及Key-File database.skey。

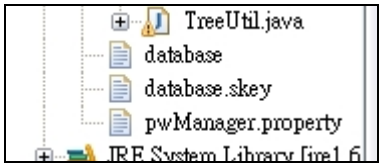


圖 5 - 7　　File 一覽

圖 5 - 8、圖 5 - 9爲使用binary editor開啓database與database.skey的結果：

database | database.skey

```
00000000  12 35 62 5D A0 1A 09 C3  70 81 86 01 18 E1 96 0B  .5b]....p.......
00000010  EF 5C 86 CF AA F1 C5 61  52 7C 64 C1 00 31 2B BD  .\.....aR|d..1+.
00000020  C8 FF D3 38 6D D0 E3 42  64 00 07 C0 BB B0 DA C1  ...8m..Bd.......
00000030  39 E1 70 90 42 6E 95 02  53 42 D6 B4 D1 75 7C 76  9.p.Bn..SB...u|v
00000040  AA 3D 69 C4 4F 68 F9 E4  AC E0 FB FF F6 4D 87 E0  .=i.Oh.......M..
00000050  C8 FF D3 38 6D D0 E3 42  64 00 07 C0 BB B0 DA C1  ...8m..Bd.......
00000060  FF 81 F1 F1 AC 86 BB 59  87 BD F1 41 5F C7 71 19  .......Y...A_.q.
00000070  36 66 AC E0 18 94 96 55  37 A7 C7 EB BD 57 27 A0  6f.....U7....W'.
00000080  14 21 DD 3D D6 08 7B C7  45 77 FE 3B 8B 54 85 4C  .!.=..{.Ew.;.T.L
00000090  D5 4C BA F9 04 0B 03 AF  7B 8D 8F 28 9F 31 74 38  .L......{..(.1t8
000000a0  02 13 AD B7 90 EB 38 57  41 61 8A 60 AE 30 D5 BB  ......8WAa.`.0..
000000b0  A1 55 11 A5 9E 10 52 54  AE 85 B7 A8 FC 5F 01 38  .U....RT....._.8
000000c0  05 99 8E B8 49 E0 FD 7B  E7 E9 1C 84 AA C8 B6 51  ....I..{.......Q
000000d0  D9 41 AC 1D 10 65 88 F3  98 25 CC 1C B0 BB 63 C6  .A...e...%....c.
000000e0  DE 91 23 86 55 52 AF 8A  26 D6 0F B7 10 76 04 B4  ..#.UR..&....v..
000000f0  6F 51 54 6D ED DE 2A 21  48 F1 5C 6C 1B 08 9B 55  oQTm..*!H.\l...U
00000100  56 72 9F C5 7B 46 84 24  D6 BF 60 D8 2C 1B CB 37  Vr..{F.$..`.,..7
00000110  EB 77 AB 82 BF 95 67 A6  C6 1C A4 83 12 F2 06 1D  .w....g.........
00000120  EA 3D AC EF 1E 22 61 52  32 8C C3 F6 BF 52 CB D6  .=..."aR2....R..
00000130  12 97 3C E7 5A 5F DA D6  B2 CB 6B 9C 81 F1 22 7A  ..<.Z_....k..."z
00000140  94 64 50 70 AC 08 77 31  5C 35 DD 2C 18 0F ED 4C  .dPp..w1\5.,...L
00000150  40 6E FC 6B 0A DB 4C 3A  23 0A 40 7C 58 C8 BF 91  @n.k..L:#.@|X...
00000160  F0 6E 9C 27 98 1A 6F 18  54 C3 EE 4A 78 C5 95 EE  .n.'..o.T..Jx...
00000170  B4 FD 9B 7E AA E3 DD 7A  3F 9D 8A 2D A9 1F AD 6A  ...~...z?..-...j
```

圖 5 - 8　　Database 加密結果

database | database.skey

```
00000000  BA F4 66 62 C9 E7 53 67  EE 5C 9A 36 EA 30 A9 07  ..fb..Sg.\.6.0..
00000010  E1 7A A1 73 9C 2E A7 E9  DF 1C BF 41 63 BC 2C 73  .z.s.......Ac.,s
00000020  36 D4 94 AE 7B 23 C8 8A  32 D5 48 60 BA 4B BC 71  6...{#..2.H`.K.q
00000030  18 20 EC B6 35 9E 2D E2  12 D0 CB 8B 15 0A BD 40  . ..5.-........@
00000040  C1 E8 77 72 18 1A F5 F9  E7 D5 EE 5B 55 34 AD E8  ..wr.......[U4..
00000050  22 48 0A 02 D6 46 47 BA  04 C1 E6 22 6E 1D E6 DD  "H...FG...."n...
00000060  75 3D 05 6F 30 AE 7D 4D  84 7C B3 88 16 46 DC F4  u=.o0.}M.|...F..
00000070  BC DA 3E 47 7A A8 5F 5D  B1 12 8C 5B 0B C3 8B 8B  ..>Gz._]...[....
00000080  34 EA EC 79 C2 F5 ED 67  5B 78 4C 3C 9F 85 74 F8  4..y...g[xL<..t.
00000090  D1 68 46 59 49 B8 34 16  5A 2C F5 AF 62 62 D1 53  .hFYI.4.Z,..bb.S
000000a0  00 69 9D 51 1D AB 67 6D  9F 0D 3D C6 AB 4E 47 A3  .i.Q..gm..=..NG.
000000b0  E4 37 AB A1 95 21 97 3F  F8 20 DF 0C 82 EA 66 34  .7...!.?. ....f4
000000c0  43 FB DE 0E C2 F9 0E D5  0E 92 9F E2 5B 5A F0 07  C...........[Z..
000000d0  7C E6 DE D6 58 48 2D AA  34 DD D9 FA 18 4A 43 E1  |...XH-.4....JC.
000000e0  EF 60 C5 E2 55 D9 D7 05  AC 14 34 5B B4 00 97 32  .`..U.....4[...2
000000f0  DE 8E F6 C5 C7 03 50 2F  73 78 FF 6C D9 EB E4 4E  ......P/sx.l...N
00000100  B5 53 1E 7F 4D 5F FD 0E  07 F6 A8 E0 80 EF 5E F4  .S..M_........^.
00000110  78 EA 2C 0F 84 D3 17 A7  42 E3 46 7C 8F B6 38 2F  x.,.....B.F|..8/
00000120  1D 8A 6D 3F AC AC F4 80  3B 13 0E F2 53 95 D8 FA  ..m?....;...S...
00000130  A0 3A F0 2C B6 12 29 89  14 96 16 3E 35 9E 26 26  .:.,..)....>5.&&
00000140  46 81 7F C3 35 A2 15 2B  E9 13 9A AA 44 85 C2 33  F...5..+....D..3
00000150  C2 BC 73 0E F8 30 AA AB  77 FE BD F8 A0 37 F5 C0  ..s..0..w....7..
00000160  7A AC 7F 7B 20 A6 7B 66  99 3B E3 07 EB 12 0B F0  z..{ .{f.;......
00000170  B0 70 14 16 E6 24 05 DC  CE CE 2C 9A 10 35 B6 84  .p...$....,..5..
```

圖 5 - 9　　Key-File 加密結果

結果看來雜亂無章，但又如何得知檔案是否有依照我們對不同欄位的設定來做加密呢，我們在加密儲存檔案時一併把 KeyObject 中比較感興趣的欄位

print 出來：

```
MainWindow: 儲存檔案.actionPerformed
DatabaseAgenct: # Group   Encrypt: General - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: # Group   Encrypt: Internet - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: # Group   Encrypt: Portal - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: #   Entry Encrypt: NCTU - AES - HCe+dJ9l70D2b2adjUlirA==
DatabaseAgenct: #   Entry Encrypt: Google - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: #   Entry Encrypt: Yahoo - DESede - vr4jn59V0dH8Ly/X3dlGT09iVlb6n5/I
DatabaseAgenct: #   Entry Encrypt: PLASLAB - Blowfish - CzhYnBm4lvnSCpp3eWEMAA==
DatabaseAgenct: # Group   Encrypt: E-Mail - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: # Group   Encrypt: Remote Workstation - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: # Group   Encrypt: Game - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: # Group   Encrypt: Bank Account - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: # Group   Encrypt: etc. - AES - vi0fVdH8L9fdRk9iVvqfyA==
EncryptionAgency: * keyPairVec( AES ) = {Default=vi0fVdH8L9fdRk9iVvqfyA==, Hidden=HCe+dJ9l70D2b2adjUlirA==}
EncryptionAgency: * keyPairVec( Blowfish ) = {Default=vi0fVdH8L9fdRk9iVvqfyA==, Hidden=CzhYnBm4lvnSCpp3eWEMAA==}
EncryptionAgency: * keyPairVec( RC4 ) = {Default=vi0fVdH8L9fdRk9iVvqfyA==}
EncryptionAgency: * keyPairVec( DES ) = {Default=vp/RL9lPVp8=}
EncryptionAgency: * keyPairVec( DESede ) = {Default=vr4jn59V0dH8Ly/X3dlGT09iVlb6n5/I}
```

圖 5 - 10　decryptPair 與 keyPairVector 內容

圖 5 - 10中分別將decryptPair Field與keyPairVector Field內容輸出至stand out。上半部是在class DatabaseAgency中加入print要求輸出儲存檔案時decryptPair Field的內容，可清楚看到我們加密的順序以及加密的種類 (Group or Entry)、加密欄位名稱、使用Cipher以及Base64 形式的Secret Key。下半部的輸出內容是在class EncryptionAgency加入print敘述，分別以KeyLabel=SecretKey的形式輸出五種Cipher各自的keyPairVec Field，AES及Blowfish多了兩筆Key Label值爲我們新增的Hidden Label。跟上半部的decryptPair做比較，我們可以發現NCTU與PLASLAB欄位所使用的Secret Key的確如我們所設置的Hidden Label對應的Secret Key。

接著把視窗關掉後再重新執行程式，系統使用儲存的 Key-File 來開啓database，得到與儲存前相同的結果，得知本系統能夠正確的執行對於不同欄位的加密設定做混合加密。

5.3　金鑰匯出測試

在5.2小節更換Cipher與Secret Key的設置後，此節針對使用AES與Blowfish的Hidden Key Label作爲加密的NCTU與PLASLAB欄位，做隱藏的設定並且匯出金鑰；之後再使用此把金鑰對儲存的Database進行解密，並且觀察結果。

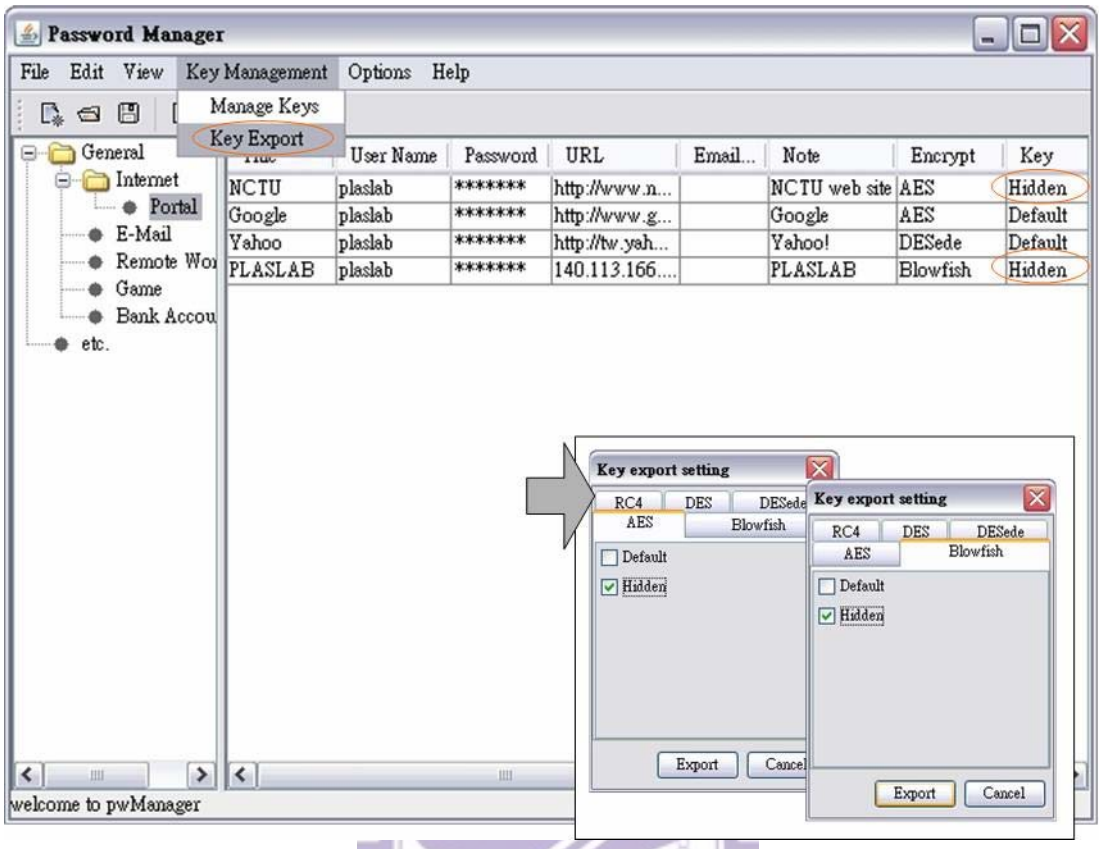


圖 5 - 11　金鑰匯出設定

要做金鑰匯出要在Key Management選單中點選Key Export的選項來做匯出的設定。如圖5 - 11，點選Key Export後會跳出圖右下方的Key export setting dialog，每一種Cipher都有所屬的Tabbed Pane，依據使用者不同的設定會有不同的勾選內容，這裡的例子是在AES與Blowfish Cipher中會多出Hidden的核取方塊。

我們希望達到隱藏 NCTU 與 PLASLAB 兩欄位的目標，因此我們分別勾選 AES 的 Hidden 選項以及 Blowfish 的 Hidden 選項。這裡勾選的用意爲在下次使用我們設定匯出的金鑰來解密檔案時，使用這些被選取設定值加密的欄位將會被系統忽略；也就是說，讀取得出的 Database 將是先前 Database 的子集合。

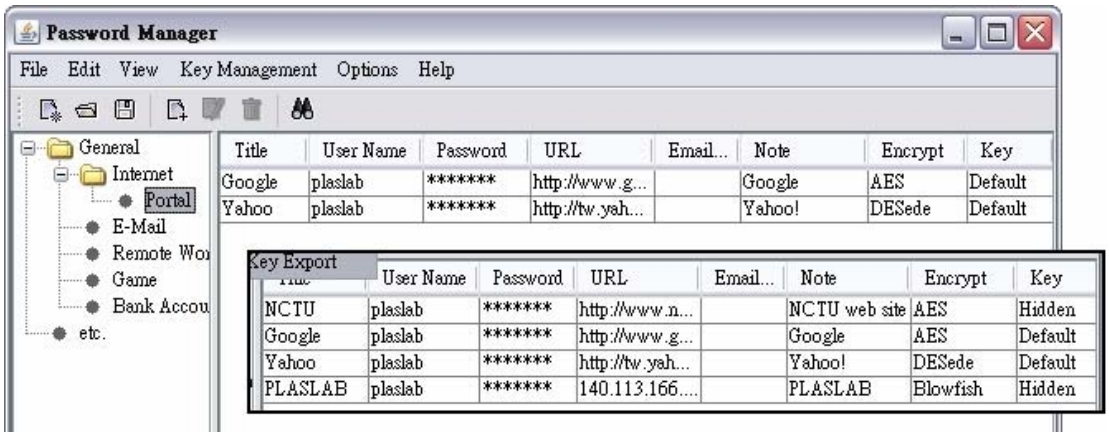


圖 5 - 12　金鑰匯出結果

我們將金鑰匯出另存爲“.skey.ex”的副檔名。接下來將視窗關閉後在開啓，得到如圖 5 - 12的結果，可以發現以AES - Hidden以及Blow - Hidden加密設定的欄位消失了。同樣的，我們再來看看匯出的金鑰內容爲何：

```
MainWindow: 儲存檔案.actionPerformed
DatabaseAgenct: # Group   Encrypt: General - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: # Group   Encrypt: Internet - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: # Group   Encrypt: Portal - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: #   Entry Encrypt: NCTU - AES - null
DatabaseAgenct: #   Entry Encrypt: Google - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: #   Entry Encrypt: Yahoo - DESede - vr4jn59V0dH8Ly/X3dlGT09iVlb6n5/I
DatabaseAgenct: #   Entry Encrypt: PLASLAB - Blowfish - null
DatabaseAgenct: # Group   Encrypt: E-Mail - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: # Group   Encrypt: Remote Workstation - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: # Group   Encrypt: Game - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: # Group   Encrypt: Bank Account - AES - vi0fVdH8L9fdRk9iVvqfyA==
DatabaseAgenct: # Group   Encrypt: etc. - AES - vi0fVdH8L9fdRk9iVvqfyA==
EncryptionAgency: * keyPairVec( AES ) = {Default=vi0fVdH8L9fdRk9iVvqfyA==, Hidden=HCe+dJ9l70D2b2adjUlirA==}
EncryptionAgency: * keyPairVec( Blowfish ) = {Default=vi0fVdH8L9fdRk9iVvqfyA==, Hidden=CzhYnBm4lvnSCpp3eWEMAA==}
EncryptionAgency: * keyPairVec( RC4 ) = {Default=vi0fVdH8L9fdRk9iVvqfyA==}
EncryptionAgency: * keyPairVec( DES ) = {Default=vp/RL9lPVp8=}
EncryptionAgency: * keyPairVec( DESede ) = {Default=vr4jn59V0dH8Ly/X3dlGT09iVlb6n5/I}
```

圖 5 - 13　金鑰匯出後的 decryptPair 與 keyPairVector 內容

和圖5 - 10稍做比較，在圖5 - 13中可發現匯出金鑰過程中，在我們勾選要做隱藏設定相對應的欄位，其用來解密的decryptPair Field的Secret Key值被系統吃掉重新設定爲null，而keyPairVec Field的設定則沒有改變。

藉由Key Export以null值設定decryptPair Field來隱藏選取欄位，系統在下次讀入檔案時依序使用decryptPair Field做解密時，若遇到Secret Key值爲null的欄位自動忽略不做讀入的動作。

另外一點要聲明的是，圖中我們也可看出 Group，也就是 Node 在存檔時使用 Default Cipher 與 Default Secret Key 加密；若我們將 Export Key 設定爲 Default Cipher 與 Default Key 的組合，那整個 Database 都會消失？答案是否定的，在實作 Key Export 機制時對於 Node 欄位強制使用 Default Cipher 與 Default Secret

Key 加密/解密，因此 Key Export 的設定不會對 Node 造成任何影響，Node 的加密設置只能由 User Password 與 Default Cipher 的改變來更動。

第 6 章
結論及未來展望

網路和生活越來越緊密的結合在一起，網路上是使用帳戶/密碼來識別身份，個人帳戶的保管必須小心謹慎。個人密碼管理系統提供了一個安全而且易於使用的環境來管理使用者的個人帳戶資訊。許多網路服務的註冊訊息或是日常生活中的瑣碎的密碼都可以藉由本系統安全的保存，並且透過 GUI 的設計對於帳戶資訊能夠輕鬆的加以編輯管理。另外系統還提供了網頁的連結功能，在帳戶的 URL 欄位輸入正確的網址即有如書籤的功能輕鬆連結 Internet。

Key-File 的實作將加密使用的 Key 從 Database 之中抽取出來，資料與金鑰分開管理增加了使用者資訊的安全性，畢竟加密的資料安全也依賴金鑰的安全。使用者透過自行設定唯一的一組 Password 做 Key-File 的解密；解密後的 Key-Object 才是解開 Database 真正的金鑰。

不同於一般的密碼管理系統，本系統使用五種 Cipher，並且配合 Key-File 的使用使用者能夠自行對於每一項欄位，也就是一筆帳戶資訊，設定相異的加密組合；即使使用相同的 Cipher 也可以產生多筆金鑰來對不同的欄位加密，即使檔案不慎遺失也可將傷害降到最低，所有的檔案資訊不會同時顯現在攻擊者眼前。

個人密碼管理系統在許多方面都還有可以精進的地方。除了用來安全的保存與管理帳戶資訊，也希望能夠透過簡易使用的介面改變懶散的不良使用習慣，降低使用者密碼過度類似的問題。要達到此目的我們在 User Friendly 方面有許多可以改進，像是介面的美化、欄位拖曳功能等，甚至對於不同的節點能夠提供不同型態帳戶的管理；像是 Internet 群組我們可以記錄網路相關資訊，而在 Bank Account 群組我們擁有不同的欄位可以記錄需要的資訊，取代原本所有節點都有相同形式的設定。

除了 Password 的使用，本系統可與其他 Authentication 服務更進一步的進行連結；如 IC 卡識別身份、使用指紋登入系統…等等現在越來越普及的身份認

證方式。本系統在未來亦可朝單一簽入的方向進行，存放許多的使用者帳戶資訊若能夠在登入個人密碼管理系統的同時登入了所有的網路服務，漫遊網路將可輕鬆愉快。

參考文獻

[1] Gang Chen, Ke Chen and Jinxiang Dong, **"A Database Encryption Scheme for Enhanced Security and Easy Sharing,"** Computer Supported Cooperative Work in Design, 10th International Conference on, May 2006.

[2] Scott Granneman, **"Phishing for Savvy Users,"** Security Focus, Nov. 1, 2004.

[3] Anti-Phishing Working Group, http://www.antiphishing.ort/

[4] Universal Password Manager, http://www.universalpasswordmanager.com/

[5] KisKis, http://kiskis.sourceforge.net/

[6] Password Safe, http://passwordsafe.sourceforge.net/

[7] PassReminder, http://eyecanseeyou.free.fr/passreminder_password_manager/

[8] KeePass Password Safe, http://keepass.info/

[9] The Internet Engineering Task Force, "**Base64 Content-Transfer-Encoding**," RFC 1521, Section 5.2, September 1993.

[10] The Internet Engineering Task Force, "**The MD5 Message-Digest Algorithm,**" RFC 1321, April 1992.

[11] National Institute of Standards and Technology, "**Data Encryption Standard (DES),**" FIPS 46-3, October 25, 1999.

[12] National Institute of Standards and Technology, "**Advanced Encryption Standard (AES),**" FIPS 197, November 26, 2001.

[13] National Institute of Standards and Technology, "**Secure Hash Standard (SHS)**," FIPS 180-2, August 1, 2002.

[14] B. Schneier, "**Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish),**" Fast Software Encryption, Cambridge Security Workshop Proceedings (December 1993), Springer-Verlag, 1994, pp. 191-204.

[15] David M. Geary, "**Graphic Java 2 : mastering the JFC/Swing**, 3rd ed," The Sun Microsystems Press, A Prentice Hall Title 1999.

[16] Sun Microsystem, Inc., "**Java look and feel design guidelines, 2nd ed**,"

Addison Wesley 2001.

[17] Scott Oaks, "**Java Security, 2nd ed,**" O'REILLY 2001.

[18] Jonathan Knudsen, "**Java Cryptography,**" O'REILLY 1998.

[19] David Hook, "**Beginning Cryptography with Java,**" Wiley Publishing, Inc. 2005.

[20] Rich Helton and Johennie Helton, "**Java Security Solutions,**" Wiley Publishing, Inc. 2002.

附錄 1
密碼使用習慣

◆ 投票中止於: Fri May 15 08:50:01 1998

◆ 票選題目描述: 請大家踴躍填寫:)

◆ 投票結果:

我會用生日,名字等各人資料當密碼	2780	票
我會用純英文當密碼	1286	票
我會用英文加數字當密碼	1478	票
我會用英文,數字,標點混合當密碼	1149	票
我會用純數字當密碼	1567	票
我會用打不出來的字當密碼	942	票
我通常只用1套密碼	1935	票
我會使用2-4套密碼	1214	票
我會用5種密碼以上在不同的帳號	229	票
我幾乎不會更改密碼	1520	票
我一年左右會換密碼	237	票
我幾個月會換一次密碼	230	票
我幾天就換一次密碼	103	票
只有我知道我自己的密碼	1280	票
有2人-4人知道我的密碼	590	票
有5人以上知道我的密碼	104	票
我的bbs密碼和工作站密碼一樣	476	票
我的bbs密碼和工作站密碼不同	776	票
我沒有工作站的帳號	479	票
我的密碼很難猜	626	票
我的密碼不太好猜	489	票
我的密碼中等難度	590	票
我的密碼滿好猜	684	票
我的密碼用肚臍都猜的出來	328	票
這個投票不錯:)	1551	票
這個投票很爛:(	462	票

◆ 總票數 = 23105 票