國立交通大學

資訊科學系

碩 士 論 文

基 於 儲 存 限 制 模 型 的 私 密 通 訊 系 統

Private Communication System based on

Bounded Storage Model

研 究 生：林坤杉

指導教授：曾文貴　教授

中 華 民 國　九 十 三　年 六 月

基於儲存限制模型的私密通訊系統

學生：林坤杉　　指導教授：曾文貴 博士

國立交通大學資訊科學系

摘　要

在一般的公開金鑰系統中都是基於一些困難的計算假設上(如 RSA assumption)。而限制儲存空間的模型(Bounded Storage Model)並不限制攻擊者的計算能力，它是假設限制攻擊者的儲存空間。而基於這種模型所設計出來的加密系統，如果攻擊者無法完全儲存在加密過程中所使用的隨機字串時，即使日後金鑰被洩露了，密文仍不會被解密，這樣的特性又稱為永久性的安全(Everlasting Security)。

在此我們應用了 Lu 在[10]中所用的證明，將 Dziembowski 跟 Maurer 的方法[8]其安全性做了一個延伸，使其能夠抵擋動態攻擊(adaptive attack)，即使金鑰重覆的被使用，而系統仍是安全的。並利用訊息驗證的觀念來得到一個不可捏造(non-malleable)的加密系統。最後設計並模擬了一個基於儲存限制的訊息傳送系統，並分析我們所得到的效能。

關鍵詞：限制儲存空間的模型、不可捏造、永久性的安全

Private Communication System based on Bounded Storage Model

Student：Kun-Shan Lin Advisor：Dr. Wen-Guey Tzeng

Department of Computer and Information Science

National Chiao Tung University

Abstract

In pulic-key cyptosystems are based on computational assumptions(i.e. RSA assumption). But there is no restirction of computational power in the bounded storage model. It just limited the adversary's storage. We can construct a encryption scheme by using this model. If the adversry can't store all the random bits which are used in the communication. Then he can't decrypt the ciphertext, even he get the initial key after communication. This property is so called everlasting security.

We apply the proof which was proposed by Lu[10] to extend the security of Dziembowski and Maurer's scheme[8]. Then we make the security of DM's scheme to against adaptive attack, even the same initial key is reused in communication. We also use the concept of message authentication to extend the scheme to get the non-malleable encryption system. Then we design a private communication system based on bounded storage model.

Key Word：Bounded Storage Model, Non-malleable, Everlasting Security

誌　　謝

在此感謝我的指導老師曾文貴教授，在我碩士班兩年的學習過程中，不只讓我在學業上受益良多，更在生活上以及言行上給我許多教導。此外，我要感謝口試委員，交大資工系蔡錫鈞教授和清大資工系孫宏民教授，在論文上給予我許多良好的建議和指導，讓我的論文更加完善。除此之外我要感謝實驗室同學，尚宸、照儀、振魁和佩琳的幫忙，實驗室學長成康、惠龍、學姊季穎的指導，以及實驗室學弟妹們在精神方面的鼓勵。

最後，我要感謝我的家人，不論在精神或物質上都給予我極大的支持，讓我在無後顧之憂的情況下可以順利完成學業。在此，謹以此文獻給我所有我想要感謝的人。

目　　錄

圖表目錄

表格目錄

第一章 引言

在網際網路的高度發展下，使用者經常會利用網路來互相傳送訊息。而在一個開放性的網路環境下，想要得到一個安全的傳送訊息方式一直是密碼學上研究的課題。不論是私密金鑰加密系統(Private Key Encryption System)或公開金鑰加密系統(Public Key Encryption System)，在金鑰被洩露的情況下都無法再保持密文的私密性。如果能夠擁有在金鑰被洩露的情況下而密文仍無法被解密的特性，這是相當令人著迷的。

在一般的公開金鑰系統中都是建構在一些困難的計算假設上(如 RSA assumption)。因此 Maurer 在 92 年提出了不同以往的假設，也就是限制儲存空間的模型(Bounded Storage Model)，在此模型中主要是限制攻擊者的儲存空間，並沒有去限制攻擊者的計算能力。基於這種模型所設計出來的加密系統，若攻擊者無法完全儲存在加密過程中所使用的隨機字串時，那麼即使在日後金鑰被洩露了，而之前的密文仍無法被解密。這樣的特性又稱為永久性的安全(Everlasting Security)。在此我們應用了 Lu 在[10]中所提的證明，將 Dziembowski 跟 Maurer 的方法其安全性做了一個延伸，使其能夠抵擋動態攻擊(adaptive attack)，即使金鑰重覆的被使用，系統仍是安全的。並利用訊息驗證的觀念來達到一個不可捏造(non-malleable)的的加密系統。然後在一般的網路上設計模擬了一個基於儲存限制的訊息傳送系統。

第一節 研究動機

有很多人常會使用 ICQ、MSN 等軟體與他人溝通，而這些軟體多數都是以明文(Plaintext)的方式在網路上傳送，對於一個惡意的攻擊者就可以在網路上竊聽你所傳送的訊息。因此如何在一個開放性的環境下得到安全的秘密通訊方式，

這一直是在密碼學裡一個很基礎的問題。安全的秘密通訊其主要目標，是令傳送者與接收者兩者間的通訊內容不會被攻擊者所得知。

過去的研究中，各種加密演算法被提了出來 (如 DES、AES、RSA、ElGamal 等)。但這些加密方法只要攻擊者儲存了加密過的密文，之後若能得到金鑰，那就能夠將密文解密。因此這些演算法都無法提供永久性的安全。針對 DES 而言已經可以利用特別設計的硬體在幾天內破解了，所以過去如果有攻擊者仍儲存著當時的密文，現在就能夠將之解密而得到機密的資訊了。而對於公開金鑰系統的安全性都是基於一個很難的計算假設(unproven assumption)上才是安全的，但這樣的假設只適用於一般的電腦，如果將相同的問題移到量子計算的電腦上，那這樣將會使得 RSA 假設變的不適用了。因此如果將來量子技術能夠有突破性的發展而量子電腦能夠普及的話，那麼安全性基於因數分解(Factoring)與離散對數(Discrete logarithm)的系統將不再安全了(如 RSA)。

如上所述，對於電腦系統與量子計算的的迅速發展，可能使一些基於公開金鑰的加密方式變的不安全。而且對於密鑰洩露後是否能夠保持著密文的私密性，這種令人嚮往的安全性使得這方面的研究相當重要。因此永久性安全系統的研究在近幾年來也越來越多，我們可以發現在這樣的演算法下，其所使用的假設是不同於過往基於一些很難的計算假設，他是基於限制攻擊者儲存空間的假設下所發展出來的。它不限制攻擊者的計算能力，所以其安全性要滿足理論安全(information-theoretic secrecy)。

第二節 研究目標與成果

限制儲存空間的模型提供了一種不同方向的安全性假設，而我們也可以針對下面幾個重點來研究。一個安全性基於限制儲存空間的系統需要包含以下幾個部份(Alice 跟 Bob 為使用者)：

1. Alice 與 Bob 間必須先共同擁有一把短的私密金鑰 *initial key K*。

2. 一個可以不停的傳送大量公開隨機字串的主機 $R \xleftarrow{random} \{0,1\}^t$. ($t$ > 攻擊者儲存空間)。

3. 一個金鑰衍生函數(*derivation function*)，其輸入是隨機字串與初始金鑰，輸出是與平均分佈(uniform distribution)成統計上鄰近(statistically close)。

4. 使用 one-time pad 這種擁有理論安全的加密方式，將衍生函數所衍生出來的金鑰視為加密金鑰來進行加密。

而金鑰衍生函數的安全性，在於攻擊者無法儲存完整的隨機字串，所以即使在日後得到初始金鑰，也無法做出相同的輸出結果。

本篇論文主要是研究各種永久性安全的演算法，我們希望能夠對各種演算法都能夠熟悉。對於兩種不同的演算法 Ding-Rabin 的方法與 Dziembowski-Maurer 的方法，在此我們研究了後者的方法並將其安全性做了一個延伸使其能夠抵擋動態攻擊，使這樣的系統其初始金鑰能夠重覆使用而系統仍是安全的。之後利用 *HMAC* 來對密文做驗證，並達到一個不可捏造(non-malleable)的性質。最後在一般的網路上設計並模擬了一個基於儲存限制的訊息傳送系統。而我們利用特別設計的資料結構與協定，使得在我們所設計的系統裡不需要做時間同步的機制，並且詳細分析在現有的網路裡所實作出來的效能。

第三節 各章節介紹

在第二章裡我們將會介紹所需要的相關背景，對限制儲存模型(Bounded Storage Model)的定義做一個詳細的介紹。並介紹兩個基於儲存限制的方法及其他相關研究。然後在第三章對接下來會用到的密碼學定義，做一個詳細的描述。攻擊模型與訊息驗證的觀念也會在這裡介紹說明。第四章我們對 Dziembowski 跟 Maurer 的安全性做一個延伸，使得即使金鑰被重覆使用，而系統仍是安全的。並且利用訊息驗證的觀念來加強原本的系統，使其能夠達到不可捏造的特性。第五章對我們實作的系統做一個詳細的描述，並分析其效能。然後是總結。

第二章 相關研究

在這個章節中我們會介紹限制儲存的基本概念與相關研究。在第一節會先對限制儲存模型(Bounded Storage Model)的定義做一個詳細的介紹。對於基於儲存空間的加密系統，我們會使用一種名為 one-time pad 的加密方法。而它是一種對稱式金鑰的系統，在第二節中我們會對這些做詳細的介紹。在第三節會介紹兩個基於限制儲存模型的加密方法 Ding-Rabin 與 Dziembowski–Maurer 的加密方法，然後介紹其他基於儲存限制的研究。

第一節 限制儲存空間的模型

一般的公開金鑰的加密方式是基於計算上很難的問題上，因此在這樣的系統中都會限制了攻擊者的計算能力。但是在限制儲存空間的模型(Bounded Storage Model)下，它並不是基於計算上很難的問題上，所以並不需要去限制攻擊者的任何計算能力。而它唯一限制的只有限制攻擊者 Eve 的儲存空間而已，這樣的設定就是一種限制儲存空間的假設。而這樣的觀點最早是在 1992 年由 Maurer 在[12]所提出來的，他假設攻擊者 Eve 在任何時間所能夠使用的儲存空間，是一個大量並且固定大小的空間(如 1 terabyte)，且並沒有對 Eve 的計算能力做任何的限制，但他所描述的攻擊者是只能夠儲存隨機字串中原本的位元，攻擊者不能利用任何的儲存函數來計算隨機字串得到計算過後的位元，所以之後也定義了一個一般性的儲存限制假設，使得攻擊者能夠利用任意的函數來計算隨機字串，只要其所輸出的長度是小於隨機字串就可以了。而這樣也使得攻擊者的能力又更強了，之後的相關研究也都是基於這樣的一般性儲存限制假設。

我們可以利用這樣的模型來建構出一個安全的加密系統，證明其安全性是基於理論安全(information-theoretically-secure)，而不是基於一些難解的問題(如因數

分解、離散對數問題)。在加解密系統中傳送者與接收者需要先共享一把金鑰，並且能夠存取一隨機字串(random bits)，而這個隨機字串是必須比攻擊者所能儲存的空間來要來的大，假設隨機位元為R而攻擊者的儲存空間為V，攻擊者可以利用任何的儲存函數來對隨機字串做運算，其輸出的長度必須限制在 $|V| < |R|$。之後即使攻擊者得到傳送方與接收方一開始所共享的金鑰，仍然無法攻擊此加解密系統。

在加解密的過程中需要大量的隨機字串(random bits)，當然要傳遞大量的隨機位元(ex 2^{50})利用硬體的實作是一個比較有效率的方式(如利用衛星來傳送)，但本篇論文是以目前的網路環境來模擬實作一個實驗性的加解密系統，並分析在現有網路環境下所得到的效能。

第二節 對稱式金鑰系統

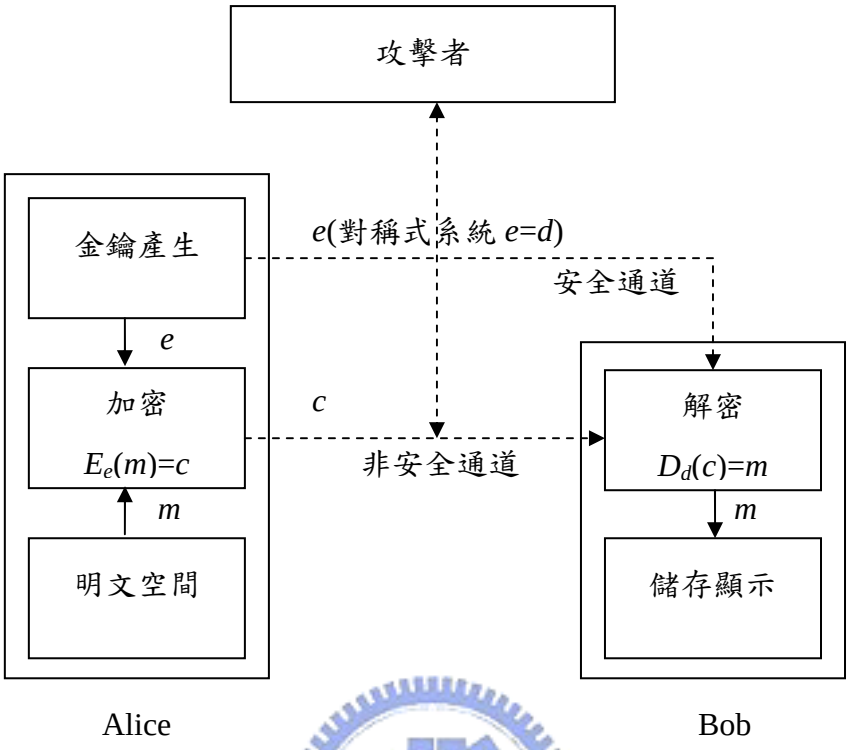
為了要達到一個安全的溝通通道，我們常會利用一些加密的演算法來加密訊息以達到一個安全的溝通。加解密的系統大致可分為兩大類，對稱式金鑰系統與非對稱式金鑰系統(symmetric-key system and asymmetric-key system)，而在本篇中所使用到的，主要是以對稱式金鑰的加密系統為主，來達到一個安全的溝通管道，在此我們會對對稱式金鑰系統做一個詳細的定義，而我們在基於儲存限制的加密系統中，主要是利用 one-time pad 這樣的加密方法，因此我們也會詳細介紹 one-time pad 這個加解密系統。

■ 對稱式金鑰系統(Symmetric-key System)

考慮一個加解密系統主要包含兩個部份，加密及解密演算法 $\{E_e : e \in \mathcal{K}\}, \{D_d : d \in \mathcal{K}\}$[1]，對稱式的系統主要是定義在加密用的金鑰與解密用的

[1] $\mathcal{K}$是一個金鑰空間(key space)

金鑰是相同的情況下，也就是當$e=d$，圖表 1 為一對稱式系統的安全通道：



圖表 1 Alice 與 Bob 利用加密來達到安全通道

Shannon 在 1949 年討論了密碼系統的安全性[16]，他主要考慮了兩個問題：

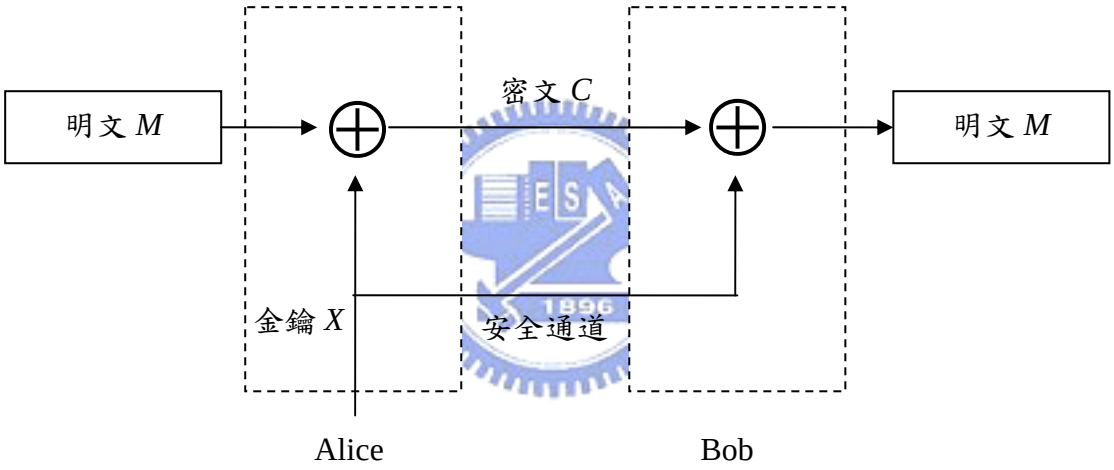
1. 當攻擊者有無限的計算能力(Computational power)時，對密文加以分析時，一個密碼系統能有多強的安全性?
2. 當攻擊者在有限的時間及計算能力的限制下對密文做分析，那這樣子一個密碼系統是否足夠安全呢?

我們考慮當攻擊者不論得到多少的密文 C，並且對密文以無限制的時間與計算能力來加以分析。如果在這樣的情況下攻擊者所能夠得到的明文資訊，其結果跟直接猜測明文是一樣的話；那麼如果一個系統能夠達到這樣的安全性，我們就稱它是理論安全，或絕對安全(perfect security)。而 Shannon 也說明了要達到這樣的安全性，所使用的加密金鑰的長度必須大於或等於明文的長度。

■ **one-time pad 系統**

這樣的系統是一種擁有理論安全的一種系統，所以這樣的加密方法，可以用

來抵擋一個不被限制計算能力的攻擊者。在 one-time pad 這種系統當中，每次要加密並傳送一個訊息 M 時，傳送者與接收者之間必須先共同擁有一把隨機的私密金鑰 X，稱為 *one-time pad*，金鑰與明文彼此之間是互相獨立的，且 $|X| = |M|$ ($|\cdot|$表示其二進位位元長度)。當傳送者要加密訊息 M 時，只需計算 $C = M \oplus X$ ($\oplus$ 為互斥或的運算 XOR)，當接收者收到密文 C 想要對其解密時，只要計算 $C \oplus X = M \oplus X \oplus X = M$ 就可以得到訊息 M 了，如圖表 2 所示。值得注意的是每個 X 只能對一個訊息加密，若要加密其他訊息必須再共享另一個 X。因此對於每次想要加密訊息就必須重新溝通出一把新的 X，這在現實的實作上是 one-time pad 系統的一個缺點。



圖表 2 one-time pad system

我們考慮上面所提到的 one-time pad 系統，雖然其運算是一個很簡單的互斥或運算，但對於攻擊者而言想要分析密文 $C(|C|=n)$，會發現所有 2^n 可能的明文都有可能經過金鑰 X 的加密而成此密文，所以很清楚的它是一個有理論安全的系統，這樣的系統雖可達到最高安全性，但是因為明文長度通常會很長，如此要獲得一個跟明文長度相同的或更長的金鑰是一大難題。因此這個系統如果要加密長的文件時就必須先共享一把夠長的 one-time pad，所以在一般是被認為不適用於實際場合，但在接下來的章節裡我們所介紹到的方法，都是藉由一把短的初始金鑰，從一個長的但是公開的隨機字串中，萃取出一把擁有足夠雜亂性的 one-time

pad X，所以我們也可以視這些方法在 one-time pad 系統下，為一個產生共享金鑰的協定。

第三節 兩個基於儲存限制的方法

■ Ding and Rabin 的方法

Ding 跟 Rabin 在[9]所提出的方法是 Alice 與 Bob 間先建立一把共同分享的初始金鑰$\vec{s}$，其中每個s_i都是一個 k 元素的向量，向量中的值是介於 0 到 t 之間的數，每次加密都以長度 b-bit 的訊息為單位來做加密，因此這把共享的金鑰其實是由bk 個索引所組成的，所以這把共享金鑰所需的長度為$bk\log_2 t$。

接下來若要加密一長度 b 的訊息 M，Alice 與 Bob 必須去聽取一個長度為 t 公開的隨機字串 R，並且根據金鑰$\vec{s}$中的索引來記錄相關的位元，計算出所要的 *one time pad* $X=(X_1,...,X_b)=(\oplus_{j=1}^{k}R[\sigma_{1j}],...,\oplus_{j=1}^{k}R[\sigma_{bj}])$，Alice 對 M 做加密運算 $C=M\oplus X$，將 C 傳送給 Bob，當 Bob 想解密時只要計算$M=C\oplus X$。

這個系統其安全性能夠抵擋 Chosen Ciphertext Attack(詳細定義在第三章)。當共享金鑰$\vec{s}$被建立後，在這樣的攻擊下能夠重覆使用達$2^{O(k)}$次，即使之前所用過的 *one time pad* 被攻擊者得知，共享的金鑰$\vec{s}$也可以繼續在下一次的加解密步驟中再被使用；即使$\vec{s}$在之後被攻擊者得知，因為當時所使用的隨機字串已經不能被存取，且攻擊者無法完整儲存當時所使用的隨機字串，因此在這之前所加密的文件仍然是安全的，這就是所謂的永久性安全(Everlasting Security)。

Message Block $M \in \{0,1\}^b$

Secret key $\vec{s} = (s_1,...,s_b), s_i = (\sigma_{i1},...,\sigma_{ik}) \in \mathcal{S} = \{1,...,t\}^k$. ($k$ is security parameter)

Public random string $R \xleftarrow{random} \{0,1\}^t$.
Both Alice and Bob listen while R is being broadcast.

1. Alice and Bob store $R[\sigma_{ij}]\ \forall 1 \le i \le b, 1 \le j \le t$.
2. *for* i=1 to b *do*

   Alice and Bob set $X_i = f(s_i, R) = \oplus_{j=1}^{k} R[\sigma_{ij}]$
3. Alice and Bob set

   $X = \vec{f}(\vec{s}, R) = (X_1,..., X_b) = (f(s_1, R),..., f(s_b, R)) = (\oplus_{j=1}^{k} R[\sigma_{1j}],..., \oplus_{j=1}^{k} R[\sigma_{bj}])$
4. Alice encrypts $C = X \oplus M$, and sends C to Bob
5. Bob decrypts $M = C \oplus X$

圖表 3 Ding and Rabin 的方法

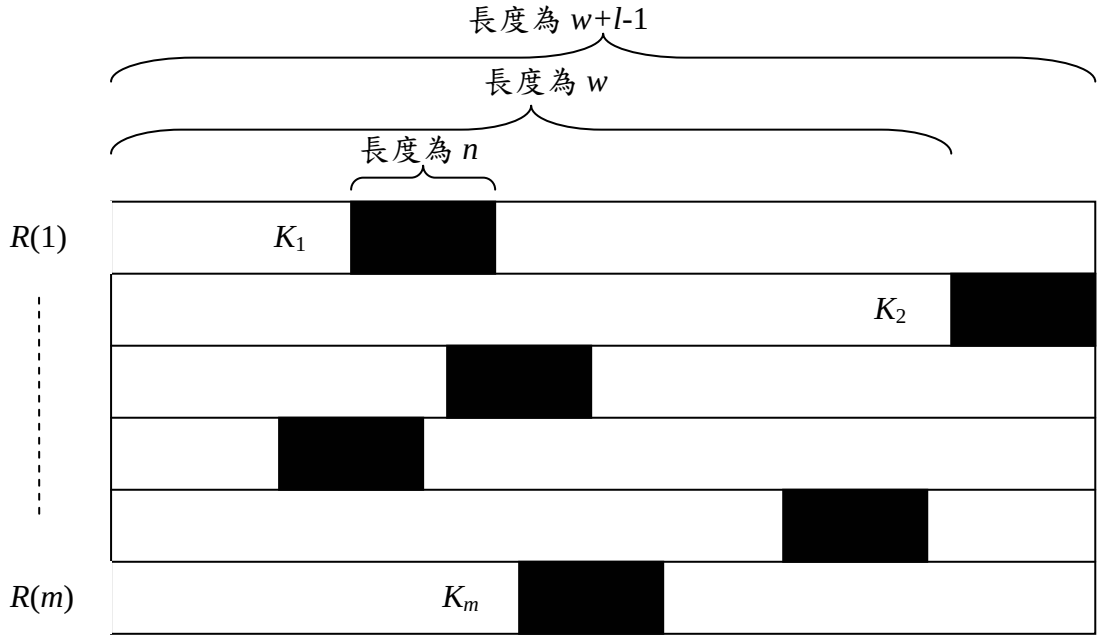
■ **Dziembowski and Maurer 的方法**

Dziembowski 跟 Maurer 在[8]的方法只有說明如何將一把短的初始金鑰(initial key) K，經過他們所提的方法來得到一把長的衍生金鑰(derived key) X，其系統的安全性為 X 是一個與平均分佈成統計上鄰近的亂數。

在此我們介紹他們是如何產生出衍生金鑰的，首先Alice與Bob間先建立一把共同分享的金鑰 $K = (K_1,..., K_m) \in \{1,..., w\}^m$，將隨機字串$R$視為$m$個區塊組成的，$K_i$主要是用來記住每個區塊$R(i)$要從第幾個位元開始的起始索引，這把共享的初始金鑰是由 m 個起始索引所組成的，每個索引需要 $\log_2 w$ 的位元來儲存，因此這把初始金鑰所需的長度為 $m \log_2 w$。

接下來聽取一個公開的隨機字串 $R \in \mathcal{R} = \{0,1\}^t$，如圖表 4 所示，根據金鑰 K 中的起始索引將其後 n 個位元記錄起來，針對 R 中的區塊 $R(1),\ldots,R(m)$，由 m 個起始索引所引導的長度為 n 的字串，將這 m 個字串做互斥或的運算後就可計算

出所需要的 *one time pad* $X = f(R,K) = (\oplus_{i=1}^{m} R(i)[K_i],...,\oplus_{i=1}^{m} R(i)[K_i + n - 1])$。



圖表 4 Dziembowski 跟 Maurer 的方法

■ **兩種方法的比較**

我們對上面所提的兩種方法做一個比較，如圖表 5 所示：

	Ding and Rabin	Dziembowski and Maurer
攻擊者儲存能力	$0.166 \times \lvert R \rvert$	$0.08 \times \lvert R \rvert$
金鑰長度	$bk \log_2 t$ (與訊息長度相關)	$m \log_2 w^2$(小於訊息長度)
安全性	可抵擋動態攻擊者，達到 CCA 的安全程度	沒有證明可以抵檔動態攻擊
重覆使用金鑰	可，系統還是安全	不可，沒有證明

圖表 5 DR 跟 DM 方法的比較

在安全性方面 Dziembowski 跟 Maurer 並沒有證明他們可不可以抵擋動態攻擊，但在 Ding and Rabin 的方法裡對於初始金鑰的長度與訊息長度是成正比的，

[2] $m \log_2 w \le n$ *n*為產生的衍生金鑰，也可視為可加密的訊息長度

且我們要傳遞 *b*-bit 的訊息確得先共享一把比訊息長的初始金鑰。所以我們會對 Dziembowski 跟 Maurer 的方法進一步的研究，使其能夠達到 CCA 的安全性，並且能夠重覆使用初始金鑰，而系統還是安全的。

第四節 其他相關研究

雖然儲存限制的模型在[12]就定義出來了，Maurer 所提出的是一個私密金鑰的溝通協定，且其安全性的證明並不是在一般性的儲存限制模型。在 Maurer 所提的系統中攻擊者所能儲存的位元必須是跟公開隨機字串的的位元是一樣的，並不能夠儲存經過任意運算後的位元。因此在之後 Cachin 跟 Maurer 在[4]提出的系統中，攻擊者可以儲存經過任意運算後的位元，只要總數不超過被限制的儲存空間即可，但在這樣的系統中所得到的結果效率並不好，為了要得到足夠的安全性，Alice 跟 Bob 必須儲存相當大量的位元(i.e. 3×10^{10})。

對於加密系統在[1][2]中也有相關研究，但是所提出的加密系統其安全性並沒有考慮動態攻擊者，而在[9][6]中所提出的加密系統中就有考慮動態攻擊者的安全性。在[10]中提出的加密系統是利用萃取器(extractor)概念，作者利用一個列表解碼(List-decoding code)來建立出一個強萃取器(strong extractor)，然後從隨機字串中得到 one-time pad。

對於[9]中所提的方法而言，如果要加密訊息必須先共享一把比訊息長的初始金鑰，但其安全性是可以抵達 CCA 的攻擊。而在[8]中所提的方法不需要共享一把比訊息長的金鑰，因此效能比較好，但是其安全性並沒有考慮 CCA 的攻擊，也沒有對 one-time pad 的捏造性做改善的動作。

其它方面的研究如金鑰分配(key agreement)的系統，Maurer 也有提出許多相關的研究[13][14][16]。而 Ding 在[5][6]中提出一個基於儲存限制的模糊傳送系統(oblivious transfer)，而訊息驗證(message authentication)與不可捏造(non-malleable)的加密系統在[6][9]中也有相關的研究。

第三章 背景與基本定義

在本章節我們定義一些密碼學上的基本定義，還有之後證明會使用到的基本定理。在第二節會介紹攻擊者模型，詳細說明 CCA/CPA 的攻擊方式。其他所需要的相關背景也會在後面有詳細的說明。

第一節 基本定義

定義一：語意上的安全(*semantically secure*)

一多項式 $p(k)$ 對於所有的安全參數 k，則若一密碼系統 $CS=(E,D,G)$，對於 $\forall M_0,M_1\in\{0,1\}^m$ 且 $(e,d)\leftarrow G(1^k)$，攻擊者任何的解密演算法 A 都滿足下列式子，則稱它是 $p(k)-semantically\ secure$

$$|\Pr[A(E(e,M_0)=1]-\Pr[A(E(e,M_1))=1]|<\frac{1}{p(k)}$$

對於一個沒有計算限制的攻擊者，一個理論安全(*informational-theoretically secure*)的私密金鑰加密系統，表示其所能夠達到的安全程度是 *semantically secure* 且滿足下式 $|\Pr[A(E(e,M_0)=1]-\Pr[A(E(e,M_1))=1]|<2^{-O(k)}$

定義二：統計距離(*statistical distance*)

對於兩個隨機變數(random variable)X、Y 而且他們的基礎樣本空間(common sample space)為 S，X 與 Y 之間的統計距離(*statistical distance*)定義如下：

$$\begin{aligned}dist(X,Y)&=\frac{1}{2}\sum_{s\in S}|\Pr[X=s]-\Pr[Y=s]|\\&=\max_{T\subseteq S}\{\Pr[X\in T]-\Pr[Y\in T]\}\end{aligned}$$

定理一：對於任意的隨機變數 X，Y，Z 而且他們的基楚樣本空間為 S

$$dist(X,Y)\le dist(X,Z)+dist(Z,Y)$$

定理二：[10]如果有一機率分佈 A，它與機率分佈 B 跟 B'彼此是互相獨立的，那麼對於任何的函數 f 他們的基礎樣本空間為 S,Q，我們得到以下的式子：

$$dist(f(A,B),f(A,B'))=dist(B,B')$$

定義三：多項式不可分辨(*polynomial indistinguishable*)

任意兩個機率分佈 X_0,X_1，且對於任何機率的多項式時間杜林機器(probabilistic polynomial-time Turing machine, PPTM) D，若$dist(D(X_0),D(X_1))$是可忽略的情況下，對於每個多項式$p(n)$都存在n_0使得 $n\geq n_0$ 如果滿足下面式子，我們說它們是多項式不可分辨

$$|\Pr[D(X_0)=1]-\Pr[D(X_1)=1]|\leq 1/p(n)$$
$$or\ dist(D(X_0),D(X_1))\text{ 是可忽略的}$$

定義四：統計上鄰近、近亂數(*statistically close*、*ε-random*)

如果兩機率分佈X跟Y，若其$dist(X,Y)\leq\varepsilon$(ε是一個可忽略的函數)，那我們稱X跟Y是統計上鄰近(*statistically close*)，如果Y是一個平均分佈(uniform distribution)的話，那我們稱X是一個***ε-random***的分佈。很清楚的可以發現，如果X是***ε-random***的話，那也表示X跟平均分佈是多項式不可分辨的。而對於機率分佈X_0跟平均分佈U之間，如果X_0與U兩者間是多項式不可分辨的話，那我們稱這個分佈X_0是近亂數。

定理三：[6]令 X 是一個機率分佈在$\{0,1\}^m$下，且是一個近亂數，如果一個 *one-time pad* 系統用 X 來當作其所使用的 *one-time pad*，則此系統對 m-bit 的訊息擁有語意上的安全。

第二節 攻擊者模型

對於密碼系統的安全性而言，大多數都是考慮攻擊者的攻擊模型。如果一個密碼系統能夠抵擋特定的攻擊者模型，我們就相信它是安全的，接下來我們會介

紹兩種攻擊者模型。

♦選擇性密文攻擊(Chosen Ciphertext Attacks, CCA)：

選擇性密文攻擊對於任一加解密系統$\mathcal{CS}=(G,E,D)$，若一個攻擊者$\mathcal{AD}$要攻擊此系統，$\mathcal{AD}$可以去查詢一個解密資料庫(Decryption oracle)$\mathcal{O}_d(\cdot)$。詳細的攻擊方式分成兩個步驟：

步驟一：$\mathcal{AD}$可以對解密資料庫$\mathcal{O}_d(\cdot)$做l次的查詢動作，對於第i次的查詢可以根據之前所做的i-1 次查詢所得到的訊息，來決定第i次要詢問解密資料庫的密文C_i，然後會得到相對應於C_i的明文$M_i=D(d, C_i)$(d是一把解密金鑰)。

步驟二：$\mathcal{AD}$會被給予另一密文C，這個密文C是沒有詢問過解密資料庫$O_d(\cdot)$的，$\mathcal{AD}$必須根據步驟一所作的l次查詢得到的資訊$(M_1,C_1)\ldots(M_l,C_l)$，來猜出相對應於密文C的明文M。

♦選擇性明文攻擊(Chosen Plaintext Attacks, CPA)：

選擇性明文攻擊對於任一加解密系統$\mathcal{CS}=(G,E,D)$，若一個攻擊者$\mathcal{AD}$要攻擊此系統，$\mathcal{AD}$可以去查詢一個加密資料庫(Encryption oracle)$\mathcal{O}_e(\cdot)$。詳細的攻擊方式與選擇性密文攻擊一樣分成兩個步驟：

步驟一：$\mathcal{AD}$可以對加密資料庫$\mathcal{O}_e(\cdot)$做l次的查詢動作，對於第i次的查詢可以根據之前所做的i-1 次查詢所得到的訊息，來決定第i次要詢問加密資料庫的明文M_i，然後會得到相對應的密文$C_i=D(e, M_i)$(e是一把加密金鑰)。

步驟二：跟選擇性密文攻擊的步驟二一樣。$\mathcal{AD}$被給予另一密文C，$\mathcal{AD}$必須根據步驟一得到的資訊$(M_1,C_1)\ldots(M_l,C_l)$，來猜出相對應於密文C的明文M。

對於上面所提到的兩種攻擊方式，可以很清楚的發現在公開金鑰的加密系統中，CPA 的方式是比較沒意義的。因為公開金鑰是大家都知道的，所以可以對任

何的訊息做加密，因此通常都是討論能夠抵擋 CCA 的攻擊。對於私密金鑰的加密系統就必須分開討論。但在本論文會用到的方法中，可以很清楚的發現 CCA 跟 CPA 是等價的。

第三節 訊息驗證

當我們在傳送一份訊息 M 時，如果不希望訊息的內容被修改，通常會對訊息做一些運算得到其相對應的訊息驗證碼 r(message authentication code)，所以一個訊息驗證系統若想要會先對 M 產生相對應的 r，然後傳送(M,r)，當接收者收到後針對訊息 M 做相同的運算。藉由 r 的正確性來判斷訊息傳送的過程裡有沒有被第三者修改。一般在公開金鑰系統裡想要達到訊息驗證的性質我們可以利用數位簽章的技術來做到此性質。但在此我們討論的是在私密金鑰系統下安全的訊息驗證。

定義五：在此定義一個私密金鑰的訊息驗證系統，這樣的系統主要有三個部份 $MAS=(Gen,Mac,Ver)$，Gen 為一個產生私密金鑰的演算法，Mac 為一個產生訊息驗證碼的演算法，Ver 為驗證演算法，詳細如下：

- $Gen(1^k)$，輸入為一長度為k的隨機字串(k為系統安全參數)，其輸出為私密金鑰s
- $Mac(M,s)$，輸入為任意的訊息 $M\in\{0,1\}^*$，私密金鑰 s 是由 Gen 所產生的，其輸出是一個訊息驗證碼 $r=Mac(M,s)$
- $Ver(M,s,r)$，其輸入為訊息 M、私密金鑰 s，與要驗證的驗證碼 r。當驗證正確時輸出 1，表示(M,r)是有效的，反之輸出 0

$$Ver(M,s,r)=\begin{cases}1 & if\ \ r=Mac(M,s)\\ 0 & if\ \ r\neq Mac(M,s)\end{cases}$$

在一個私密金鑰系統，當傳送者與接收者共享一把私密金鑰 s，傳送方利用演算法 Mac 計算出 $r=Mac(M,s)$；接收方也是擁有相同的私密金鑰 s，所以也可

以計算出 $Mac(M,s)$，藉由驗證演算法 $Ver(M,s,r)$輸出的結果來判斷訊息有沒有被修改。如果這樣的系統是安全的，表示任一攻擊者$\mathcal{AD}$在沒有私密金鑰 s 的情況下，無法做出任一訊息 $\tilde{M}$ 其相對應的驗證碼 $\tilde{r} = Mac(\tilde{M},s)$，使得 $Ver(\tilde{M},s,\tilde{r}) = 1$。

第四節 不可捏造的加密方法

一個公開金鑰加密系統中如果有不可捏造的性質(non-malleable)，表示任一攻擊者$\mathcal{AD}$在得到一密文 C 的情況下；$\mathcal{AD}$要由 C 計算出另一個有效的密文 C'，在計算時間為一多項式時間的情況下是不可行的。而這樣的定義也可以延伸到私密金鑰的系統裡，根據[7]中提到要使一私密金鑰擁有不可捏造的特性，比在公開金鑰上的設定要來的簡單而且有效率。在公開金鑰的設定下是大家都知道公開金鑰的，所以已經有部份的資訊可以被攻擊者來使用。而在私密金鑰的設定上，只有溝通雙方知道共享金鑰。因為私密金鑰只有使用者雙方知道，所以攻擊者必須在只知道密文資訊的情況下，偽造出一份新的可以通過驗證的密文，才算攻擊成功。我們會在下面說明在 one-time pad 加密系統中，如何捏造一個有效的密文。

對於我們的不可捏造的加密系統，其主要的概念是利用訊息驗證的方式，對密文做驗證。而對於一不可捏造的加密系統，其安全性也可以利用上一節所介紹過的 CCA/CPA 攻擊者模型。之後我們也會證明我們的不可捏造的加密系統是能夠抵擋 CCA/CPA 的。

one-time pad 系統的捏造性：在 one-time pad 的系統中我們可以很清楚的發現到，它是一個可捏造的系統，如果我們有一密文 $C = M \oplus X$ ，針對此密文 C 我們做下面的運算：

$$C' = C \oplus Z = (M \oplus X) \oplus Z = (M \oplus Z) \oplus X = M' \oplus X$$

很明顯的經由一個簡單的互斥或運算後我們可以得到 C' 為 M' 用 X 所加密出來的密文。

第四章 一個基於儲存限制的加密系統

接下來我們會將 Dziembowski 跟 Maurer 的安全性做一個延伸，主要的想法是利用 Lu 在[10]中使用的證明。然後再搭配上訊息驗證的觀念，來達到一個不可捏造的加解密系統，我們會證明其安全性是可以抵擋動態攻擊者(adaptive adversary)。而在這樣的系統中，雙方所共有的初始金鑰能夠在一安全參數 l 下，重覆使用 l 次，且加解密系統仍是安全的。

第一節 DM 加密系統安全性的延伸

- **基本符號定義**

  K：初始金鑰，可視為一有m元素的向量，每個元素值小於w，用於*derivation function* 中來萃取出有足夠雜亂性[3]的 one-time pad，$K=(K_1,...,K_m)\in\mathcal{K}=\{1,...,w\}^m$

  X：經由*derivation function*，所計算出的one-time pad，長度為n

  R：公開的隨機字串，其長度為t，$R\in\mathcal{R}=\{0,1\}^t$，可將其視為$m$個短隨機字串所組成的，$R=(R(1),...,R(m))$，而每個$R(i)$的長度為$w+n-1$

  ext：是一個*derivation function*，其輸入是隨機字串與初始金鑰，輸出是one-time pad，$ext(\mathcal{R},\mathcal{K})\to\mathcal{X}$

  V：攻擊者所能儲存的相關資訊，其長度為s，$V\in\mathcal{V}$

  h：攻擊者所用的儲存函數，其輸入為隨機字串，輸出為一長度為s的字串，$h(\mathcal{R})\to\mathcal{V}$

  U、U_2：個別為相對於one-time pad $\mathcal{X}$與初始金鑰$\mathcal{K}$的平均分佈。

[3] 與平均分佈成統計上鄰近(statistically close)

■ **DM 演算法說明**

Dziembowski 跟 Maurer 的方法說明了，如何將一把短的初始金鑰(initial key) K，經過他們所提的方法來得到一把長的衍生金鑰(derived key) X。在其研究中，X 是一個與平均分佈成統計上鄰近的亂數。很明顯的可以知道，如果 X 是與平均分佈成統計上鄰近的亂數，那麼就存在一個語意上安全的 one-time pad 加密系統。我們將 DM 的加密系統以演算法的方式呈現如下，接下來我們將 Lu 的證明應用到 DM 加密系統中，使其安全性延伸至能夠抵抗 CCA/CPA 的攻擊，使得相同的初始金鑰能夠重覆使用。

演算法 一：DM with one-time pad

Initial：*A public random bits R is composed by m blocks.*

$$R = (R(1),...,R(m)),\ |R(i)| = w+n-1,\ |R| = m(w+n-1) = t.$$

Alice and Bob have：*initial key* $K = (K_1,...,K_m) \in \{1,...,w\}^m$.

Alice and Bob listen while R is being broadcast.

1. *Alice and Bob store* $R(i)[K_i],...,R(i)[K_i+n-1] \quad 1 \le i \le m$
2. *Alice and Bob compute X*

   for $i = 0$ *to* n-1 ***do***

   $$X[i] = \oplus_{i=1}^{m} R(i)[K_i + i]$$

   end for
3. *Alice compute* $C = X \oplus M$ *then transfer to Bob*
4. *Bob compute* $M = C \oplus X$

■ **安全性的討論**

假設攻擊者可以儲存任何的位元只要其長度小於 $|R|$。存在一個函數

$h:\mathcal{R}\rightarrow\mathcal{V}$，攻擊者可以利用 h 來儲存有關於 R 的任何位元：$V=h(R)$。

如果攻擊者擁有關於隨機字串 R 的資訊 V，且在之後得到了初始金鑰 K。得到 K 後攻擊者不能再存取 R 的情況下，對 X 做分析。如果在這種情形下所得到的機率分佈期望值是很小的值(如 $2^{-m/2}$)，那利用這樣的 X 所做的 one-time pad 系統就是安全的。我們定義一隨機變數 $\beta(v,k)=dist(\langle X\mid V=v,K=k\rangle,\langle U\rangle)$。

引理一： [8]如果 w、m、n，能滿足 $m\log_2 w\le n$ 且 $w>100$，限制攻擊者的儲存空間約為 $s=0.08t-1.5m(n+1)$，則 $\beta(V,K)$ 的期望值會有如下之關係式：

$E[\beta(V,K)]\le n2^{-m/2}$

安全性的考量在[8]中，並沒有在之前所定義的攻擊模型下(CCA/CPA)。只說明攻擊者在得到所能儲存的資訊 V 及之後可以得到初始金鑰 K 的情況下，系統所製造出的 X 是與平均分佈成統計上鄰近的亂數。接下來我們會利用[10]中的證明，用數學歸納法的方式將其安全性證明延伸至能夠抵擋 CCA/CPA 攻擊。

首先我們說明攻擊者的攻擊策略，假設$R_1,R_2\ldots$是個別被利用在密文$C_1,C_2\ldots$與明文$M_1,M_2,\ldots$之間的隨機字串。攻擊者$\mathcal{AD}$可以存取隨機字串$R_1,R_2,\ldots$並且觀察密文與明文$(C_1,M_1),(C_2,M_2),\ldots$。針對每一組的密文與明文可以根據選擇性密文攻擊或選擇性明文攻擊所提供的解密/加密資料庫來獲得(C_i,M_i)，而攻擊者得到(C_i,M_i)後就可以計算出 $ext(R_i,K)=X_i=C_i\oplus M_i$。

因為R_i每次都是獨立選取的，且在CCA與CPA模型中，攻擊者可以得到的資訊只有(M_i,C_i)，雖然也可以利用既有的(M_i,C_i)經過計算得到 $ext(R_i,K)=X_i=M_i\oplus C_i(1\le i\le l)$。但對於在CCA與CPA中攻擊者而言，其所得到的資訊是一樣的。因此對於我們使用的方法而言，這兩種安全性是等價的。

依照之前所定義的攻擊者模型 CCA/CPA，在此說明攻擊者$\mathcal{AD}$的攻擊方式，可分為兩個步驟：

步驟一：$\mathcal{AD}$可以做l次的查詢動作，對於第i回合$\mathcal{AD}$都可以存取該回合所使

用的隨機字串R_i，攻擊者的攻擊順序為：先聽取隨機字串R_i，$\mathcal{AD}$可以利用一個儲存函數h_i來儲存s位元，接下來將R_i、上一次所儲存的資訊V_{i-1}及之前所使用的one-time pad $X_{[i-1]}=(X_1,\ldots,X_{i-1})$，當做是$h_i$的輸入。則在第$i$個回合攻擊者可以得到新的資訊為$V_i = h_i(R_i, V_{i-1}, X_{[i-1]})$。之後攻擊者無法再聽取隨機字串，但可以利用查詢資料庫獲得第i回合的密文及明文(C_i,M_i)計算出相對應的X_i。

步驟二：在第l+1 回合時，$\mathcal{AD}$會被給予另一密文C，這個密文C是沒有被詢問過的。$\mathcal{AD}$可以藉由存取這一回合的隨機字串R_{l+1}與步驟一得到的資訊，再利用函數h_{l+1}來得到新的儲存資訊$V_{l+1} = h_{l+1}(R_{l+1}, V_l, X_{[l]})$。得到新的儲存資訊後，攻擊者無法存取$R_{l+1}$，之後攻擊者可以得到初始金鑰$K$，並企圖利用這些資訊來猜出相對應於密文$C$的明文$M$。

我們將前l次的$X_{[l]}$也考慮進去，重新定義隨機變數$\beta(\bullet)$如下

$$\beta(v,k,x_{[l]}) = dist(\langle X_{l+1} \mid V_{l+1}=v, K=k, X_{[l]}=x_{[l]} \rangle, \langle U \rangle)$$

定理四：如果w、m、n，能滿足$m\log_2 w \le n$且$w>100$，限制攻擊者的儲存空間約為$s = 0.08t - 1.5m(n+1)$，則在任何的儲存函數$h_i:(R_i,V_{i-1},X_{[i-1]}) \to V_i$下$\beta(V_{l+1},K,X_{[l]})$的期望值會有如下之關係式：$E[\beta(V_{l+1},K,X_{[l]})] \le (l+1)n2^{-m/2}$

證明：

我們利用數學規納法的方式來證明$E[\beta(V_i,K,X_{[i-1]})] \le i \times n \times 2^{-m/2}$，規納在$i$上。

Basic step：i=1➔V_1=$h_1(R_1)$根據引理一$E[\beta(V_1,K)] \le n \times 2^{-m/2}$

Induction hypothesis：對i=l時成立$E[\beta(V_l,K,X_{[l-1]})] \le l \times n \times 2^{-m/2}$

Statement to be shown in induction step：考慮 $i=l+1$ 時

$$\beta(V_{l+1},K,X_{[l]}) = dist\left(\left\langle X_{l+1}\,|\,V_{l+1},K,X_{[l]}\right\rangle,\left\langle U\right\rangle\right)$$

$$\le dist\left(\left\langle X_{l+1}\,|\,V_{l+1},K,X_{[l]}\right\rangle,\left\langle X_{l+1}'\,|\,V_{l+1},K',X_{[l]}\right\rangle\right)$$
$$+ dist\left(\left\langle X_{l+1}'\,|\,V_{l+1},K',X_{[l]}\right\rangle,\left\langle U\right\rangle\right)\ \ (where\ \ X_{l+1}' = ext(K',R_{l+1}))$$

$$\Rightarrow E[\beta(V_{l+1},K,X_{[l]})]$$

$$\le E\left[dist\left(\left\langle X_{l+1}\,|\,V_{l+1},K,X_{[l]}\right\rangle,\left\langle X_{l+1}'\,|\,V_{l+1},K',X_{[l]}\right\rangle\right)\right]$$
$$+ E\left[dist\left(\left\langle X_{l+1}'\,|\,V_{l+1},K',X_{[l]}\right\rangle,\left\langle U\right\rangle\right)\right]$$

K'是從平均分佈中選取的，與$X_{[l]}$是互相獨立的。接下來我們將會分成兩個部分來討論。在第一個部分等同是討論在經過l個回合後K是否還是擁有足夠的雜亂性(randomness)。第二部分則是在第$l+1$回合時換了另一把新的初始金鑰K'，然後討論在這樣的情況下所製造出來的one-time pad X_{l+1}是否還是能夠擁有足夠的雜亂性。我們先考慮在第一部分的情況

$$E\left[dist\left(\left\langle X_{l+1}\,|\,V_{l+1},K,X_{[l]}\right\rangle,\left\langle X_{l+1}'\,|\,V_{l+1},K',X_{[l]}\right\rangle\right)\right]$$

$$= \sum_{V_{l+1}}\sum_{K}\sum_{X_{[l]}} \Pr\left[V_{l+1}=v,K=k,X_{[l]}=x_{[l]}\right]$$
$$\times dist\left(\left\langle X_{l+1}\,|\,V_{l+1}=v,K=k,X_{[l]}=x_{[l]}\right\rangle,\left\langle X_{l+1}'\,|\,V_{l+1}=v,K'=k,X_{[l]}=x_{[l]}\right\rangle\right)$$

$$= \sum_{V_{l+1}}\sum_{K}\sum_{X_{[l]}} \Pr[V_{l+1}=v,K=k,X_{[l]}=x_{[l]}]$$
$$\times\frac{1}{2}\sum_{X_{l+1}}\left|\Pr\left[X_{l+1}=x\,|\,V_{l+1}=v,K=k,X_{[l]}=x_{[l]}\right]-\Pr\left[X_{l+1}'=x\,|\,V_{l+1}=v,K'=k,X_{[l]}=x_{[l]}\right]\right|$$

$$= \frac{1}{2}\sum_{V_{l+1}}\sum_{K}\sum_{X_{[l]}}\sum_{X_{l+1}}|\Pr[V_{l+1}=v,K=k,X_{[l]}=x_{[l]}]$$
$$\times\left(\Pr\left[X_{l+1}=x\,|\,V_{l+1}=v,K=k,X_{[l]}=x_{[l]}\right]-\Pr\left[X_{l+1}'=x\,|\,V_{l+1}=v,K'=k,X_{[l]}=x_{[l]}\right]\right)|$$

$$= \frac{1}{2}\sum_{V_{l+1}}\sum_{K}\sum_{X_{[l]}}\sum_{X_{l+1}}|\Pr[V_{l+1}=v,K=k,X_{[l]}=x_{[l]},X_{l+1}=x]$$
$$-\Pr[V_{l+1}=v,K'=k,X_{[l]}=x_{[l]},X_{l+1}'=x]|$$

$$= dist\left(\left\langle V_{l+1},K,X_{[l]},X_{l+1}\right\rangle,\left\langle V_{l+1},K',X_{[l]},X_{l+1}'\right\rangle\right)$$

$$= dist\left(\left\langle h_{l+1}(R_{l+1}, V_l, X_{[l]}), K, X_{[l]}, ext(K, R_{l+1})\right\rangle, \left\langle h_{l+1}(R_{l+1}, V_l, X_{[l]}), K', X_{[l]}, ext(K', R_{l+1})\right\rangle\right)$$

由於我們的R_{l+1}是跟V_l、$X_{[l]}$、K、K'是互相獨立的，可以利用定理二所提到的結果，得到下面的式子

$$dist\left(\left\langle h_{l+1}(R_{l+1}, V_l, X_{[l]}), K, X_{[l]}, ext(K, R_{l+1})\right\rangle, \left\langle h_{l+1}(R_{l+1}, V_l, X_{[l]}), K', X_{[l]}, ext(K', R_{l+1})\right\rangle\right)$$

$$= dist\left(\left\langle V_l, K, X_{[l]}\right\rangle, \left\langle V_l, K', X_{[l]}\right\rangle\right)$$

$$= dist\left(\left\langle V_l, K, X_{[l]}\right\rangle, \left\langle V_l, U_2, X_{[l]}\right\rangle\right) \because K' \text{ is selected from uniform distribution}$$

$$= dist\left(\left\langle V_l, X_{[l-1]}, K, X_l\right\rangle, \left\langle V_l, X_{[l-1]}, U_2, X_l\right\rangle\right)$$

$$\le dist\left(\left\langle V_l, X_{[l-1]}, K, X_l\right\rangle, \left\langle V_l, X_{[l-1]}, U_2, U\right\rangle\right)$$

$$= \sum_{V_l}\sum_{K}\sum_{X_{[l-1]}} \Pr\left[V_l = v, K = k, X_{[l-1]} = x_{[l-1]}\right]$$
$$\times dist\left(\left\langle X_l \mid V_l = v, K = k, X_{[l-1]} = x_{[l-1]}\right\rangle, \left\langle U \mid V_l = v, U_2 = k, X_{[l-1]} = x_{[l-1]}\right\rangle\right)$$

$$= E[\beta(V_l, K, X_{[l-1]})]$$

$$\le l \times n \times 2^{-m/2} \qquad (1)$$

接下來讓我們詳細討論在第二個部份所發生的情況，$E\left[dist\left(\left\langle X_{l+1}' \mid V_{l+1}, K', X_{[l]}\right\rangle, \left\langle U\right\rangle\right)\right]$，這個部分就是在第 l+1 回合時換了一把新的金鑰，所以根據引理一這個值會被限制在$n \times 2^{-m/2}$之下

$$E\left[dist\left(\left\langle X_{l+1}' \mid V_{l+1}, K', X_{[l]}\right\rangle, \left\langle U\right\rangle\right)\right]$$

$$= E[\beta(V_{l+1}, K', X_{[l]})]$$

$$\le n \times 2^{-m/2} \qquad (2)$$

因此由(1)、(2)我們可以得知

$$E\left[\beta\left(V_{l+1}, K, X_{[l]}\right)\right] \le l \times n \times 2^{-m/2} + n \times 2^{-m/2} = (l+1) \times n \times 2^{-m/2}$$

故得證。

所以定理四說明了，即使初始金鑰在之後洩露了，依據演算法一所產生出來的 X 仍然是一個與平均分佈成統計上鄰近的亂數。也可以發現，因為是在 CCA/CPA 的模型下證明，所以允許攻擊者做 l 次的查詢，而在第 l+1 回合所得到的 X 仍是與平均分佈成統計上鄰近的亂數。所以很清楚的知道同樣的一把初始金鑰可以重覆使用達 l 次，而系統仍然是安全的。因此若一 one-time pad 系統使用這樣的 X 來加密，這樣的 one-time pad 系統會是語意上安全的，且能夠重覆使用初始金鑰。

第二節 不可捏造的加密系統

接下來我們考慮訊息驗證的系統，並將其概念應用到演算法一，達到一個不可捏造的加密系統。在一些公開金鑰的系統中可以利用數位簽章的方式，很容易的達到訊息驗證的效果。而在對稱式金鑰的系統(symmetric-key system)中也可以利用 MAC 的技術來達到。我們利用訊息驗證的技術對密文做驗證，使得在傳遞的過程中不會被第三者修改。

第一項 基於金鑰式赫序函數的訊息驗證

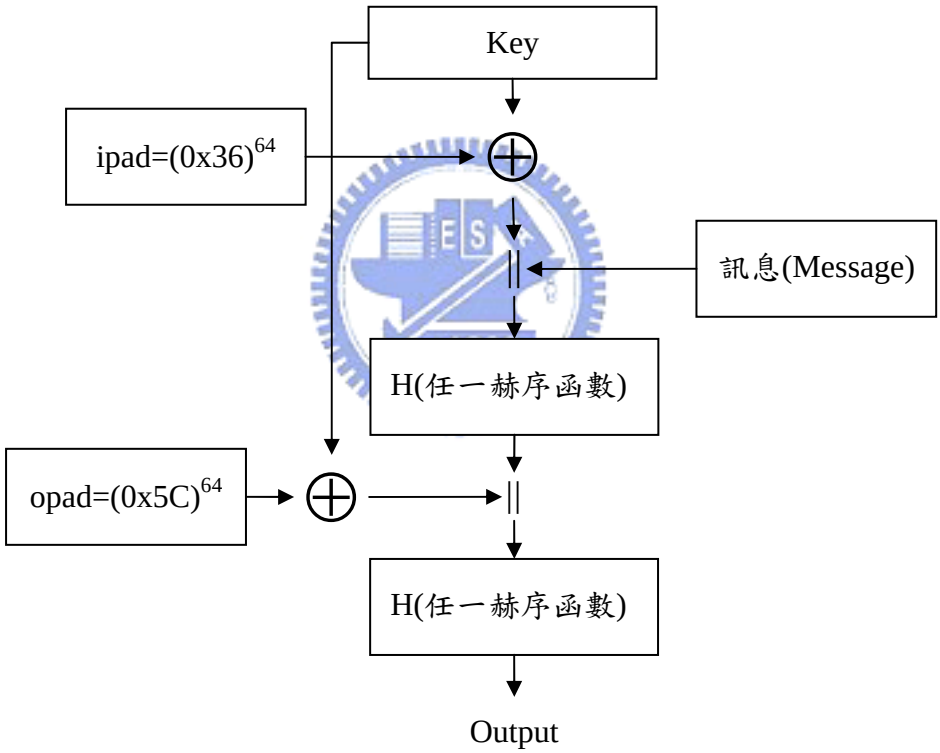
金鑰式赫序函數(keyed hash function)，其主要目地就是要用來產生訊息驗證碼(message authentication code)。與一般的赫序函數的不同，主要在於它會牽涉到一把私密的金鑰，因此只有擁有金鑰的使用者能夠產生相同的輸出。這樣的特性剛好很適合用在訊息驗證上面。我們在此介紹一種用赫序函數來產生 MAC 的演算法，”HMAC”(Keyed-Hash Message Authentication Code)。這個方法可以利用私密金鑰來計算出訊息驗證碼，並且擁有一些特性：

- 可以利用一些常見的赫序函數來製造，且已有一些赫序函數在軟體上有不錯的效能，而且也有一些免費的開放原始碼(Open Source)
- 利用既有的赫序函數來製造，其效能並不會有太大的影響，仍然有跟使

用的赫序函數差不多的效能。

- 這樣的製作過程所產生的訊息驗證，其安全性分析是基於所使用的赫序函數，而且如果有更安全的赫序函數被發展出來，也能夠很輕易的替換使用，達到更好安全程度的訊息驗證。

*HMAC*其演算法中ipad=$(0x36)^{64}$與opad=$(0x5C)^{64}$為兩個 64 位元組的定值，*H*為任何的赫序函數(如SHA-1,MD5)，對於*HMAC*而言可以將*H*視為一個黑盒子(black box)，直接像是呼叫子函式一樣的呼叫使用*H*，定義其函數如下：$HMAC(Message, Key) = H((Key \oplus opad) \| H((Key \oplus ipad) \| Message))$，詳細過程如圖表 6。



圖表 6 HMAC

密碼學上的赫序函數都是將一個任意長的輸入字串，對應到一個較短的且固定長度的輸出字串上。而一般的建構方式如SHA-1,MD5 等函數其設計都是要滿足抗碰撞的性質(*collision-resistance*)。此外，密碼學上的赫序函數通常都會有雜亂的性質(randomness)。如果一個赫序函數是"random"的，那其定義域的值就會

平均的對應到值域上。假設輸出的長度為|*T*|的話，那麼兩個隨機的輸入對應到相同的輸出的機率會是 $2^{-|T|}$。我們在下面對抗碰撞赫序函數做一個詳細的定義。

定義六：抗碰撞赫序函數(*collision-resistance hash function*)：對於一個函數集合 $\mathcal{H}=\left\{h_n:\{0,1\}^* \to \{0,1\}^n\right\}$，如果滿足下面三個條件，我們稱它擁有抗碰撞(collision resistance)的性質：

1. 在多項式時間內就可計算出所對應的值
2. *Hard to invert*：對於任何的 PPTM *M*，在任何多項式 *p*(*n*)下，滿足

$$\Pr_{y\in\{0,1\}^n}\left[M\left(1^n,y\right)\in h_n^{-1}(y)\right]<1/\,p(n)$$

3. *Hard to find collision*：對於任何的 PPTM *M*，在任何多項式 *p*(*n*)，滿足

$$\Pr\left[M\left(1^n\right)=(x_1,x_2),h_n(x_1)=h_n(x_2)\right]<1/\,p(n)$$

在[3]中詳細說明了 *HMAC* 安全性。因為 *HMAC* 主要是利用一般的赫序函數來建構的，因此若要考慮其安全性是需要在一個適當的假設之下。假設若存在一抗碰撞赫序函數，就可以做出一個安全的 *HMAC*。概略的說來，要攻擊 *HMAC* 等於是要偽造一個合法驗證碼，如果攻擊者能夠做到下面兩件事，就可以成功偽造驗證碼：

1. 攻擊者在赫序函數中找到碰撞的狀況
2. 攻擊者偽造一個有效的驗證碼

第一個狀況跟假設矛盾，而第二個狀況若發生則表示赫序函數的雜亂性(randomness)變弱了。但是因為一般在設計赫序函數都會考慮其雜亂性，所以這個狀況也會矛盾。

我們利用這樣的訊息驗證方式，將其應用到我們所使用的加密方法上，得到一個擁有不可捏造的加密系統。我們假設在 *HMAC* 中所使用的是一個性質好的赫序函數，其擁有抗碰撞的特性也擁有雜亂性。詳細的加密方式及安全性的討

論，我們將會在下一小節中討論。

第二項 擁有不可捏造的加密系統

演算法 二：Non-malleable encryption

Initial：A public random bits R is composed by m blocks.

$$R = (R(1),...,R(m)),\ |R(i)| = l + n - 1,\ |R| = m(l + n - 1).$$

Alice and Bob have：initial key K_{enc} *and* K_{auth}.

Alice and Bob listen while R is being broadcast.

5. *Alice and Bob store* $R(i)[K_{enc,i}],...,R(i)[K_{enc,i} + n - 1] \quad 1 \le i \le m$
   $R(i)[K_{auth,i}],...,R(i)[K_{auth,i} + n - 1] \quad 1 \le i \le m$

6. *Alice and Bob compute X*

   for *i* = 0 *to n*-1 ***do***

   $X_{enc}[i] = \oplus_{i=1}^{m} R(i)[K_{enc,i} + i]$

   $X_{auth}[i] = \oplus_{i=1}^{m} R(i)[K_{auth,i} + i]$

   end for

7. *Alice compute* $C = X_{enc} \oplus M$ and $r = HMAC(C, X_{auth})$ *then transfer* (*C*,*r*) *to Bob*

8. *Bob check* $r \stackrel{?}{=} HMAC(C, X_{auth})$

   if equal compute $M = C \oplus X_{enc}$

   else $\perp$

在第二章已經介紹過one-time pad系統的捏造方法，為了避免密文被偽造，我們將這樣的訊息驗證應用到DM的加密方法。在一開始要溝通的雙方必須先建立兩把初始金鑰$K_{enc} \in \mathcal{K}$、$K_{enc} \in \mathcal{K}$，所存取的隨機字串是R。利用這兩把初始金鑰來對隨機字串產生出兩組one-time pad，$X_{enc} = ext(R, K_{enc})$、$X_{auth} = ext(R, K_{auth})$。將$X_{enc}$跟之前所述一樣與訊息$M$做互斥或的運算得到$C$。為

防止密文傳遞的過程被第三者修改，我們利用X_{auth}並利用*HMAC*函數來製作驗證碼$r = HMAC(C, X_{auth})$，傳送(C,r)的組合給Bob，他必須先檢查r的正確性來判斷密文在傳遞時是否有被修改過，然後再用X_{enc}來做解密的步驟，詳細的演算法如上所示。

■ 安全性討論

這樣的系統安全性如何，我們考慮在第二章所定義的不可捏造的定義，如果攻擊者無法在沒有初始金鑰的情況下產生出一組有效的(密文-驗證碼)組合的話，那我們的系統就有不可捏造的特性。但在此我們考慮更弱的假設，如果攻擊者也能夠得到初始金鑰的狀況下系列仍是安全的。攻擊者在得到(C,r)後就不能再存取隨機字串了；直覺的想法，因為$r=HMAC(C,X_{auth})$，所以在定義四的描述下，即使K_{auth}在之後被洩露了，X_{auth}仍然是與平均分佈是統計上鄰近的。因此在K_{auth}洩露的情況下無法從(C,r)偽造出另一組有效的$(\tilde{C},\tilde{r})$。

攻擊者的攻擊策略，假設$R_1,R_2\ldots$是個別被利用在密文$(C_1,r_1),(C_2,r_2)\ldots$與明文$M_1,M_2,\ldots$之間的隨機字串，我們將$(C_i,r_i)$視為密文。攻擊者$\mathcal{AD}$可以存取隨機字串$R_1,R_2,\ldots$並且觀察密文與明文$((C_1,r_1),M_1),((C_2,r_2),M_2),\ldots$。針對每一組的密文與明文可以根據選擇性密文攻擊與選擇性明文攻擊所提供的解密/加密資料庫來獲得$((C_i,r_i),M_i)$且$r_i = HMAC(C_i, ext(R_i, K_{auth})) = HMAC(C_i, X_{auth,i})$。

依照之前所定義的攻擊者模型做一些修改，給予攻擊者每回合用來加密及訊息驗證的 one-time pad，攻擊者$\mathcal{AD}$的攻擊可分為兩個步驟：

步驟一：$\mathcal{AD}$可以做l次的查詢動作。對於第i回合$\mathcal{AD}$可以存取該回合所使用的隨機字串R_i。給予$\mathcal{AD}$隨機字串R_i、上一次所儲存的資訊V_{i-1}及之前用來加密及訊息驗證的 one-time pad $X_{enc,[i-1]}=(X_{enc,1},\ldots,X_{enc,i-1})$與$X_{auth,[i-1]}=(X_{auth,1},\ldots,X_{auth,i-1})$，$\mathcal{AD}$可以利用一個儲存函數$h_i$來儲存$s$位元的資

訊。則在第i個回合，攻擊者可以得到新的儲存資訊為$V_i = h_i(R_i, V_{i-1}, X_{enc,[i-1]}, X_{auth,[i-1]})$。之後$\mathcal{AD}$可以利用查詢資料庫獲得密文及明文$((C_i,r_i),M_i)$並給予這個回合所用的$X_{enc,i}$與$X_{auth,i}$。

步驟二：在第l+1 回合時，$\mathcal{AD}$會被給予另一密文(C,r)，這個密文是沒有被詢問過的，

$$C = M \oplus X_{enc}$$
$$r = HMAC(C, X_{auth})$$

$\mathcal{AD}$可以藉由存取這一回合的隨機字串R_{l+1}與步驟一得到的資訊，利用函數h_{l+1}來得到新的儲存資訊$V_{l+1} = h_{l+1}(R_{l+1}, V_l, X_{enc,[l]}, X_{auth,[l]})$，此後攻擊者無法存取隨機字串。然後攻擊者可以得到初始金鑰K_{enc}與K_{auth}，再利用這些資訊攻擊兩個方向：

一、猜出相對應於密文 C 的明文 M

二、產生一組新的密文與驗證碼 $(\tilde{C},\tilde{r}) = \mathcal{F}(V_{l+1}, K_{enc}, K_{auth}, C, r)$ 使得

$\tilde{C} = \tilde{M} \oplus X_{enc}$ ，$\tilde{r} = HMAC(\tilde{C}, X_{auth})$ 且 $\tilde{M} \neq M$ 。

定理五：如果 w、m、n，能滿足 $m\log_2 w \le n$ 且 $w > 100$，限制攻擊者的儲存空間約為 $s = 0.08t - 1.5m(n+1)$，則在任何的儲存函數 $h_i : (R_i, V_{i-1}, X_{enc,[i-1]}, X_{auth,[i-1]}) \to V_i$ 下，不存在任何的偽造演算法 $\mathcal{F}(V_{l+1}, K_{enc}, K_{auth}, C, r)$ 能夠偽造出有效的 $(\tilde{C},\tilde{r})$，也不存在任何的解密演算法 $\mathcal{A}(V_{l+1}, K_{enc}, K_{auth}, C, r)$ 能夠解密得到相對應於密文 C 的明文 M。

證明：

第一部分 解密得到訊息 M ：

考慮第一個攻擊方向，由於K_{enc}與K_{auth}是互相獨立的，從隨機字串中計算出的X_{enc}與X_{auth}也是互相獨立的。所以從X_{auth}無法得到任何關於X_{enc}的資訊。如果這個攻擊者能夠解密得到與訊息M相關的訊息，那可以利

用"*reduction*"的方式來證明攻擊者也可以在演算法一中得到M相關的資訊。因為在定理四中所描述的是，即使經過l回合，仍不會洩露M相關的資訊。而這樣的結果會與定理四矛盾，所以不存在這樣的解密演算法可以解密得到M，詳細*reduction*如下：

如果攻擊者能夠利用解密演算法$\mathcal{A}(V_{l+1}, K_{enc}, K_{auth}, C, r)$產生出$C$對應的明文$M$的話，就可以利用$\mathcal{A}$來攻擊演算法一，首先$\mathcal{AD}$先自己隨機選擇$K_{auth}$**步驟一：**在第$i$個回合聽取隨機字串$R_i$計算出$X_{anth,i}$ $(1 \le i \le l)$，也計算新的儲存資訊V_i。

步驟二：在第l+1 回合時$\mathcal{AD}$可以聽取R_{l+1}計算出X_{auth}與新的儲存資訊V_{l+1}，然後會得到密文C與K_{enc}，之後無法再存取隨機字串。接下來利用X_{auth}算出$r = HMAC(C, X_{auth})$，並將(C,r)、V_{l+1}、K_{enc}、K_{auth}給$\mathcal{A}$，如果$\mathcal{A}$能夠成功解密回M，那麼$\mathcal{AD}$也成功地攻擊演算法一了。但這樣的結果與定理四矛盾，所以不存在這樣的解密演算法。

第二部分 偽造新的密文與驗證碼：

考慮第二個攻擊方向。由不可分辨的定義得知，如果兩機率分佈是統計上鄰近(statistically close)的話，就表示他們是多項式不可分辨的。反過來說，也就是如果有分辨器可以分辨這兩個機率分佈的話，那麼他們就不是統計上鄰近。

如果攻擊者能夠利用偽造演算法$\mathcal{F}(V_{l+1}, K_{enc}, K_{auth}, C, r)$產生出一組新的密文與驗證碼$(\tilde{C}, \tilde{r})$使得$\tilde{C} = \tilde{M} \oplus X_{enc}$，$\tilde{r} = HMAC(\tilde{C}, X_{auth})$且$\tilde{M} \neq M$。那麼就可以將這個偽造演算法當成是子函式來呼叫，造出一個分辨器D來分辨X_{auth}與平均分佈。我們假設如果存在一個$\mathcal{F}$可以成功偽造的機率是($|T|$為*HMAC*輸出的長度)：

$$\Pr\left[\tilde{r} = HMAC(\tilde{C}, X_{auth})\right] \ge 2^{|T|} + l \times n \times 2^{-m/2}$$

接下我們可以利用這樣的$\mathcal{F}$來製造一分辨器D分辨X_{auth}與平均分佈，如下所述：

演算法 $D(V_{l+1}, K_{enc}, K_{auth}, Z)$: $(Z) \xleftarrow{R} \{X_{auth}, U\}$

i. 計算出 $\sigma = HMAC(C, Z)$

ii. 計算出 $(\overline{C}, \overline{r}) = \mathcal{F}(V_{l+1}, K_{enc}, K_{auth}, C, \sigma)$

iii. 如果$\overline{r} = HMAC(\overline{C}, Z)$，則輸出 1

C 為演算法 D 的一部分，我們接下來要說明這樣造出來的演算法 D，會與定理四互相矛盾。很明顯的根據演算法的設計，我們知道

$$D(V_{l+1}, K_{enc}, K_{auth}, Z) = 1 \Leftrightarrow \overline{r} = HMAC(\overline{C}, Z)$$

$(\overline{C}, \overline{r})$ 是演算法 D 在步驟(ii)藉由呼叫$\mathcal{F}$所得到的，因此他們成功的機率是一樣的

$$\Pr\left[D(V_{l+1}, K_{enc}, K_{auth}, Z) = 1\right] = \Pr\left[\overline{r} = HMAC(\overline{C}, Z)\right]$$

如果$Z = X_{anth}$根據假設

$$\Pr\left[\overline{r} = HMAC(\overline{C}, X_{auth})\right] \geq 2^{|T|} + l \times n \times 2^{-m/2}$$

如果$Z = U$，$HMAC$ 的結果等於是隨機選取的，其機率為

$$\Pr\left[\overline{r} = HMAC(\overline{C}, U)\right] = 2^{|T|}$$

因此由上面兩個機率式子可以得知

$$\begin{aligned}&\Pr\left[D(V_{l+1}, K_{enc}, K_{auth}, X_{auth}) = 1\right] - \left[D(V_{l+1}, K_{enc}, K_{auth}, U) = 1\right] \\ &= \Pr\left[\overline{r} = HMAC(\overline{C}, X_{auth})\right] - \Pr\left[\overline{r} = HMAC(\overline{C}, U)\right] \\ &\geq l \times n \times 2^{-m/2}\end{aligned}$$

所以如果存在$\mathcal{F}$能夠偽造出新的有效的$(\tilde{C}, \tilde{r})$，那麼將會與定理四所得到的結果矛盾，因此不存在$\mathcal{F}$可以成功偽造有效的$(\tilde{C}, \tilde{r})$。

第五章 設計與實作

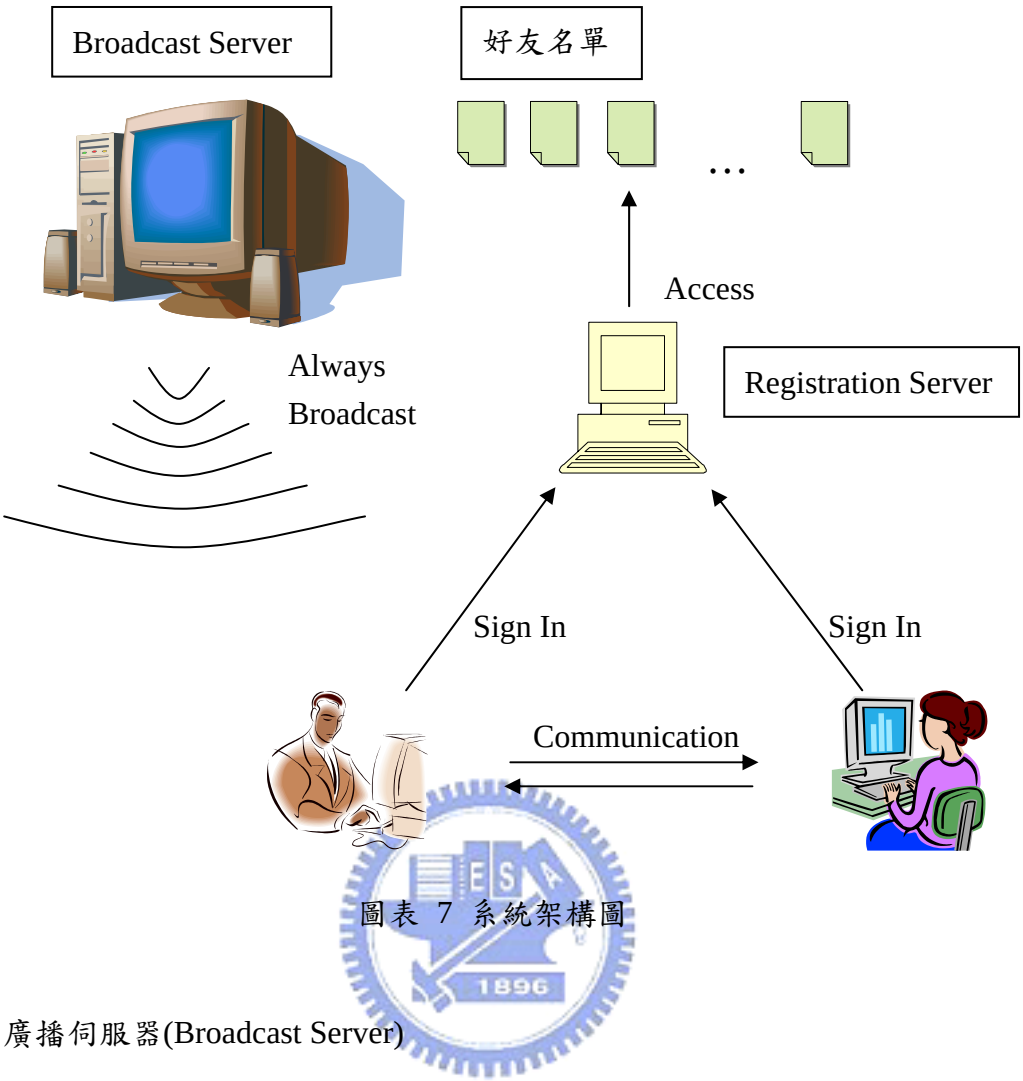
在第四章我們將 Dziembowski-Maurer 的系統安全性做了一個延伸，使其能夠達到抵擋動態攻擊者，且相同的初始金鑰在合理的安全參數下可以重覆使用。然後在之後利用了一個基於金鑰式赫序函數的訊息驗證 *HMAC*，將原本的方法中加強了一點特性，使其能夠擁有不可捏造(non-malleable)的性質。我們會依據第四章所提到的擁有不可捏造的加密方法，設計模擬一個建構在一般的乙太網路上的系統。

第一節 系統架構

系統架構概念圖如圖表 7，這個系統主要可以分成三個部分，廣播伺服器、註冊伺服器、一般使用者。廣播伺服器所使用的是一個公開的網路通道，任何人都可以去存取它所傳送的任何資料。而註冊伺服器與一般使用者之間是使用公開金鑰來做加密的安全溝通，在這個安全通道中傳遞一些註冊及登入的訊息，以防止被第三者得知相關的資訊。我們也假設這個伺服器是一個誠實的角色，它不會任意洩露使用者存在伺服器端的私密資訊。而使用者與使用者之間原本是一個公開的網路通道，但我們利用上一章節所提到的加密方式，在使用者與使用者之間利用一個基於儲存限制的加密法來達到安全的溝通。而如果攻擊者沒有完全儲存當時所使用到的隨機字串，那經過加密過的文件將會擁有一個不錯的特性，「永久性安全」。

我們要求使用者與註冊伺服器需先擁有一組公開金鑰系統的金鑰並經過一信任的機構所簽署的憑證。

接著我們將會介紹這三個角色所分配到的工作：



圖表 7 系統架構圖

- 廣播伺服器(Broadcast Server)

  這個伺服器主要的作用在於，它必須不間斷的發送隨機字串，提供使用者在溝通時所需要的大量隨機字串。

- 註冊伺服器(Registration Server)

  這個伺服器需要處理使用者註冊的事宜，並且記錄使用者密碼資訊、公開金鑰憑證、與好友名單，當有使用者登入時必須記錄該使用者目前為上線狀態，幫助使用者從他的好友名單中與目前所有的上線使用者做比對，將上線的好友資訊(包括 ID、IP address)和離線好友的名單傳送給該使用者。歸納其工作有：

  一、處理使用者向註冊伺服器註冊一個新帳號，儲存密碼資訊與憑證

  二、處理使用者登入/登出時的動作

  三、管理使用者新增好友與刪除好友，維護使用者的好友名單

- 客戶端軟體(Client Software)

  安裝在各地的電腦上，是使用者與系統真正接觸的介面。負責根據使用者提供的資訊及下達的命令，做出適當的回應。如：

  一、向註冊伺服器註冊一個新帳號

  二、登入/登出，由註冊伺服器得知線上好友的相關訊息(IP Address)，並發送上線與離線通知給線上好友

  三、新增好友與刪除好友

  四、選取想要溝通的使用者，利用公開金鑰的系統來加密所產生的共享初始金鑰，利用聽取廣播伺服器所傳送出的隨機字串與不可捏造的加密演算法，進而得到安全的溝通。

第二節 系統協定

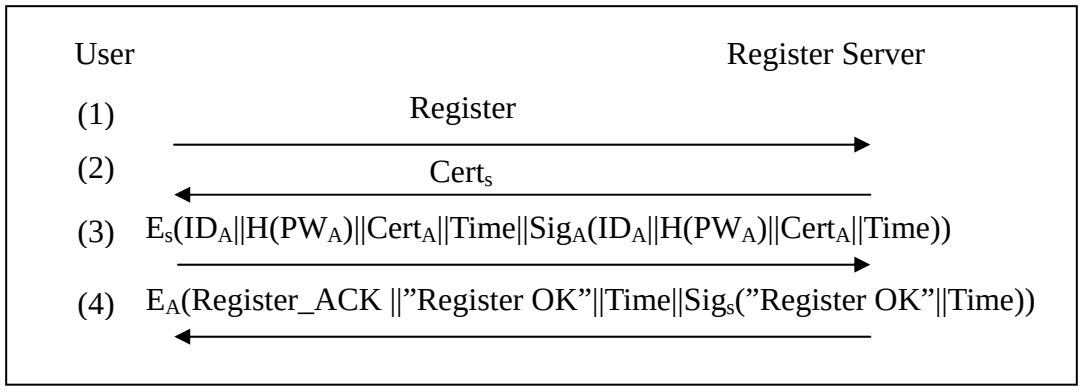
$Cert_i$：使用者i的憑證，若為s表示為註冊伺服器。

E_i：利用使用者i的公開金鑰來加密。

Sig_i：利用使用者i的私密金鑰來做簽章。

ID_i：使用者i的ID。

PW_i：使用者i的密碼。



圖表 8 註冊新帳號

- 使用者向註冊伺服器註冊一個新帳號 A

當使用者想要向伺服器註冊一個新帳號 A 的步驟如圖表 8：

(1). 使用者在一台安裝有客戶端軟體的機器上與註冊伺服器建立連線，提出註冊帳號的要求，傳送 Register。

(2). 當註冊伺服器收到使用者送來的要求，會將他的憑證傳送給使用者。

(3). 使用者得到憑證後，即可利用憑證中的公開金鑰加密訊息，將使用者的 ID、密碼相關資訊、憑證、時間戳記、對前四項訊息所做的簽章，用伺機器的公開金鑰加密起來，傳送給伺服器。

(4). 這時註冊伺服器會回傳的訊息有兩種

- Register_ACK || ”Register OK”
- Register_ACK || ”Account already exist”

當註冊伺服器解密後得到使用者的憑證，會先利用憑證來驗證簽章的正確性，再檢查時間戳記是否過期。若正確的話就回傳第一種回傳訊息，並儲存使用者的ID_A、密碼相關資訊$H(PW_A)$及使用者的憑證，反之則回傳第二種回傳訊息。

```
public class UserInfo implements Serializable{
    public String name;                    // 記錄使用者的 ID
    public String ip;                      // 記錄使用者目前的 IP Address
    public int port;                       // 記錄使用者的 Port
}
```

圖表 9 記錄使用者資料的資料結構

- 登入使用本系統

一、輸入你的帳號，傳送

$E_s(SignIn||ID_{Alice}||H(PW_{Alice})||Time||Sig_{Alice}(ID_{Alice}||H(PW_{Alice})||Time))$ 的訊息給註冊伺服器

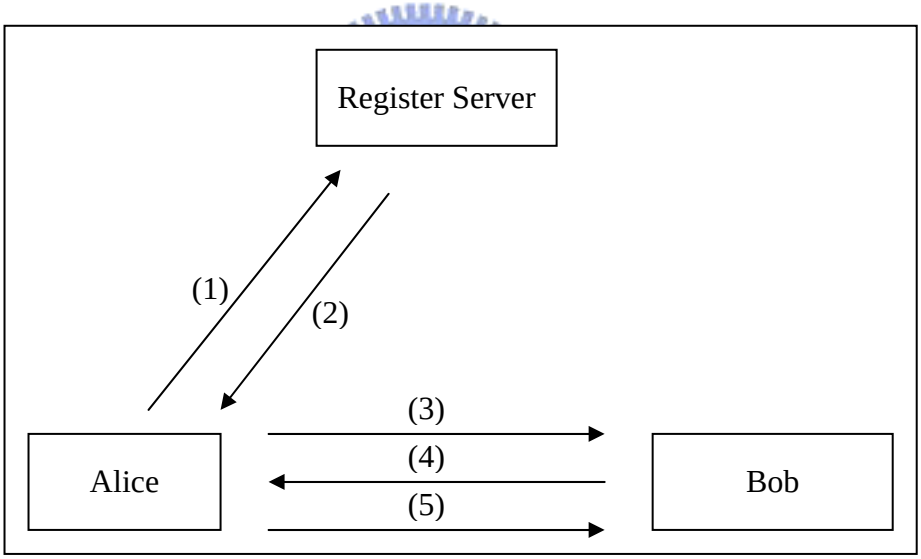
二、伺服器解密後檢查簽章的正確性，時間戳記有否過期，比對密碼相關資訊，若都正確表示登入成功，註冊伺服器會將使用者的 ID、與目前 IP Address 及 Port 記錄在我們設計的資料結構中(如圖表 9)

三、註冊伺服器會根據帳號來尋找好友名單，並且與目前連上線的使用者做比對，最後將比對後的結果加密傳回

E_{Alice} (SignIn_ACK_OK||$OnlineFriendList_{Alice}$||$OfflineFriendList_{Alice}$||Time || Sig_s($OnlineFriendList_{Alice}$||$OfflineFriendList_{Alice}$||Time))

四、當使用者解密得到資料時，對簽章驗證及時間戳記無誤後，依據$OnlineFriendList_{Alice}$的裡的IP Address資訊，傳送上線通知給你的線上好友

- 建立安全通道



圖表 10 建立安全通道

(1). Alice向註冊伺服器要求Bob的憑證，傳送

E_s(Cert_Req || ID_{Alice}|| ID_{Bob} || Time || Sig_{Alice}(ID_{Alice}||ID_{Bob} || Time))，註冊伺服器確認簽章及時戳沒問題後，傳會Bob的憑證

(2). 註冊伺服器會將Bob的憑證傳送給Alice

E_{Alice}(Cert_ACK || $Cert_{Bob}$ || Time || Sig_s($Cert_{Bob}$ || Time))，Alice檢查簽章及

時戳沒問題後，準備與Bob建立安全通道

(3). Alice傳送溝通的要求給Bob的同時，也會將自己的憑證傳送過去

$E_{Bob}(Com\_Req||ID_{Alice}||Cert_{Alice}||Time||Sig_{Alice}(ID_{Alice}||Cert_{Alice}||Time))$

(4). Bob檢查簽章及時戳沒問題後，回傳Com_ACK

$E_{Alice}(Com\_ACK||\text{“Hello”}||Time||Sig_{Bob}(\text{“Hello”}||Time))$

(5). Alice收到Com_ACK後，表示雙方之間的連線已建立。接下來雙方要溝通兩把初始金鑰，由Alice產生初始金鑰K_{enc},K_{auth}，將初始金鑰加密後傳給Bob，$E_{Bob}(Key\_Init||K_{enc}||K_{auth}||Time||Sig_{Alice}(K_{enc}||K_{auth}||Time))$。Bob解密得到內容後，確認憑證與時戳沒問題後。就可以利用初始金鑰，並聽取廣播伺服器所發送的隨機字串，計算出用來加密與用在訊息驗證的one-time pad X_{enc}跟X_{auth}，然後利用不可捏造的加密方法來做加解密的動作。

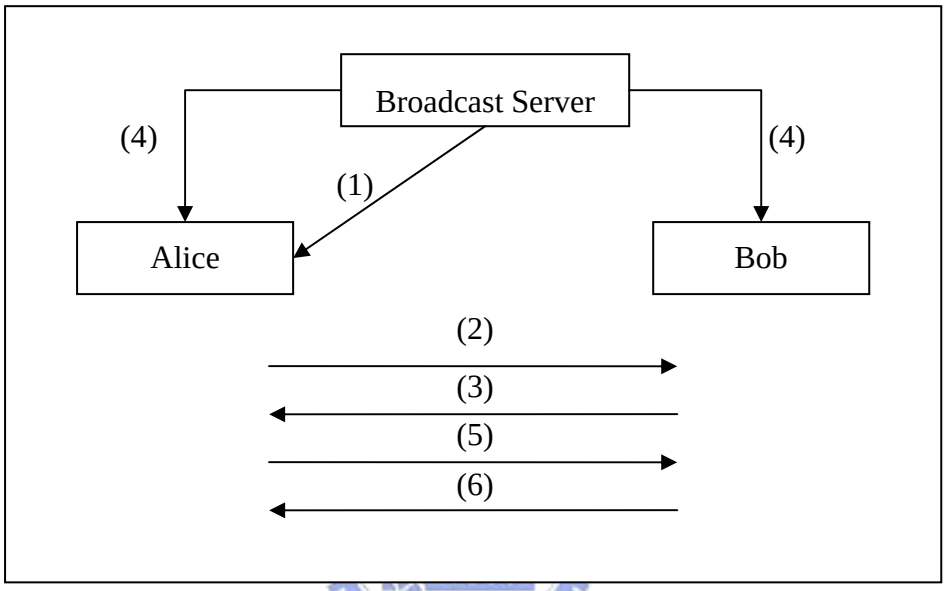
```
public class RandomData {
    public BigInteger random_bits_bignum = null;   // 記錄隨機字串
    public BigInteger index_bignum = null;         // 記錄隨機字串的索引
}
```

圖表 11 隨機位元的資料結構

聽取隨機字串的方法. 因為我們模擬的廣播方式是利用多點傳播(Multicast)，在乙太網路架構中不論是廣播還是多點傳播都是用 UDP(User Datagram Protocol)類型的封包來傳送訊息的。而 UDP 的傳送協定是一個不可靠的傳送協定，因此會有封包遺失的情況發生，所以 Alice 與 Bob 所聽取到的隨機字串，會因為某些封包的遺漏而造成收到不同的隨機字串。因此我們設計特殊的資料結構，來使我們方便解決雙方收到不同的隨機字串。

隨機字串的資料結構如圖表 11，每段隨機字串後面都伴隨著一個唯一的索引，因此由傳送訊息者去聽取目前隨機字串後面伴隨的索引值，並將其

加上 500 傳送給對方。這是因為我們在傳送索引時，也會花費時間的關係，所以為了使雙方都能從相同的索引值開始接收，因此約略的加上 500。接著雙方由得到的索引值開始接收隨機字串，當接收到一個段落時就先將索引值傳給對方，接收方經過比對後將結果回傳，雙方留下交集部份的隨機字串，反覆的做相同的動作直到隨機字串的數量足夠為止。詳細步驟如圖表 12：



圖表 12 聽取隨機字串的協定

(1). Alice 聽取目前隨機字串所伴隨的索引 *index_i*。

(2). 將 *index_i* 加上 500 後，*start_index* = *index_i* + 500，傳送 *start_index* 給 Bob。

(3). Bob 收到後回傳 ACK，*start_index_ack*。

(4). Alice 與 Bob 雙方根據 *start_index* 去接收隨機字串，將那些索引比 *start_index* 大的隨機字串儲存。

(5). Alice 將儲存的隨機字串所伴隨的索引傳串聯起來，傳送給Bob，$Index_1||Index_2||Index_3\ldots\ldots||Index_{2048}$[4]

[4] 在此我們以接收 2048 筆隨機字串為一個單位

(6). Bob 收到後，與自己所儲存的隨機字串索引做比對，將經過交集的結果以 True 或 False 表示法，回傳給 Alice。

Ex. *True*||*False*||*True*…….||*True*

- 登出本系統

一、傳送E_s(SignOut||ID_{Alice}||Time||Sig_{Alice}(ID_{Alice}||Time))的訊息給註冊伺服器。

二、註冊伺服器檢查簽章與時間戳記無誤後，將Alice從所有線上使用者的名單中刪除，並回傳

E_{Alice} (SignOut_ACK ||“SignOut OK”||Time|| Sig_s(“SignOut OK”||Time))。

三、Alice 檢查簽章與時間戳記無誤後，會依據 Online Friend List 裡的 IP Address 資訊，傳送離線通知給線上好友。

- 新增/刪除好友

一、輸入好友的帳號，傳送

E_s(AddFriendList||ID_{Alice}||ID_{friend}||Time||Sig_{Alice}(ID_{Alice}||ID_{friend}||Time)) 給註冊伺服器。

二、註冊伺服器收到後檢查簽章與時間戳記無誤後，再檢查ID_{friend}是否為已註冊過的使用者。若是一個尚未註冊的帳號，註冊伺機器回傳

E_{Alice}(AddFriendList_ACK||“not register”||Time||Sig_s(“not register”||Time))

若是一個註冊過的帳號，檢查這個帳號是否在所有線上使用者的名單中，伺服器回傳

E_{Alice}(AddFriendList_ACK||True|| $UserInfo_{friend}$ ||Time||
Sig_s(True|| $UserInfo_{friend}$ ||Time)) 若為上線

E_{Alice}(AddFriendList_ACK||False||Time||
Sig_s(False||Time)) 若為離線

，接著註冊伺機器將好友帳號寫進Alice好友名單中。

三、客戶端根據收到的傳回值來更新上線與離線的好友名單。

四、刪除好友的步驟與新增好友的一樣，只是將 Add 更改為 Delete。在此就

不詳細描述。

第三節 系統效能

在這個章節裡我們會對上面所提到的系統架構，利用一般的網路來模擬，首先說明我們的測試環境：

中央處理器 Intel 2.4-GHz Pentium 4

記憶體 256 MB

作業系統 Mircosoft Windows XP Professional

程式軟體 Borland JBuilder 8.0 with J2SDK version 1.4.2

統計與製圖軟體 Mircosoft Excel 2000

表格 1 是我們測試時所設定的環境及參數。由定理四很清楚的發現，其安全參數有一定限制在，故我們設定了這些參數但也不失去其安全性。在不同長度的隨機字串下，將 m 與產生出來的 one-time pad 長度 n 這兩者固定後，安全性就固定下來了。但還必須滿足 $m\log_2 w \le n$ 與 $w > 100$，由下表的設定看來是都滿足這兩個限制的。而在不同大小的區塊中所對應出來的初始金鑰長度 $m\log_2 w$，也可由這兩個限制發現，我們不必共享一把比訊息長度長的初始金鑰。

隨機字串長度	m	w	$m\log_2 w$	n	安全性
2MB	128	$3\times 2^{15}-1$	≈ 2176	2^{15}	$2^{15}\times 2^{-64}\approx 2^{-49}$
4MB	128	$7\times 2^{15}-1$	≈ 2304	2^{15}	$2^{15}\times 2^{-64}\approx 2^{-49}$
8MB	128	$15\times 2^{15}-1$	≈ 2432	2^{15}	$2^{15}\times 2^{-64}\approx 2^{-49}$
16MB	128	$31\times 2^{15}-1$	≈ 2560	2^{15}	$2^{15}\times 2^{-64}\approx 2^{-49}$
32MB	128	$63\times 2^{15}-1$	≈ 2688	2^{15}	$2^{15}\times 2^{-64}\approx 2^{-49}$
64MB	128	$127\times 2^{15}-1$	≈ 2816	2^{15}	$2^{15}\times 2^{-64}\approx 2^{-49}$

表格 1

我們由上面所設定的參數下，對於不同的隨機字串長度能夠傳遞多少訊息來做一個模擬。表格 2 所表示的是我們藉由調整隨機字串的長度，在不同的長度下能夠傳送多少訊息，平均每個訊息所花費的時間為何。我們測量當溝通雙方接收到相同的隨機字串長度為 2MB~64MB，每次都計算出長度 $n(2^{15})$的 one-time pad。並且以十分鐘的時間為單位來測量，所得到的效能為：在花費約十分鐘的情況下，每接受到相同的隨機字串長度為 2MB 時能產生2^{15}bits 的 one-time pad 為 79 個，平均每個 pad 花費 7.555 秒。

隨機字串長度	單位區塊 $R(i)$大小	總耗費時間	訊息總數	平均傳送時間
2MB	2^{17} bits	596.893 sec	79 個	7.555 sec
4MB	2^{18} bits	601.673 sec	31 個	19.408 sec
8MB	2^{19} bits	601.047 sec	20 個	30.052 sec
16MB	2^{20} bits	602.374 sec	11 個	54.761 sec
32MB	2^{21} bits	599.219 sec	8 個	74.902 sec
64MB	2^{22} bits	627.781 sec	4 個	156.945 sec

表格 2

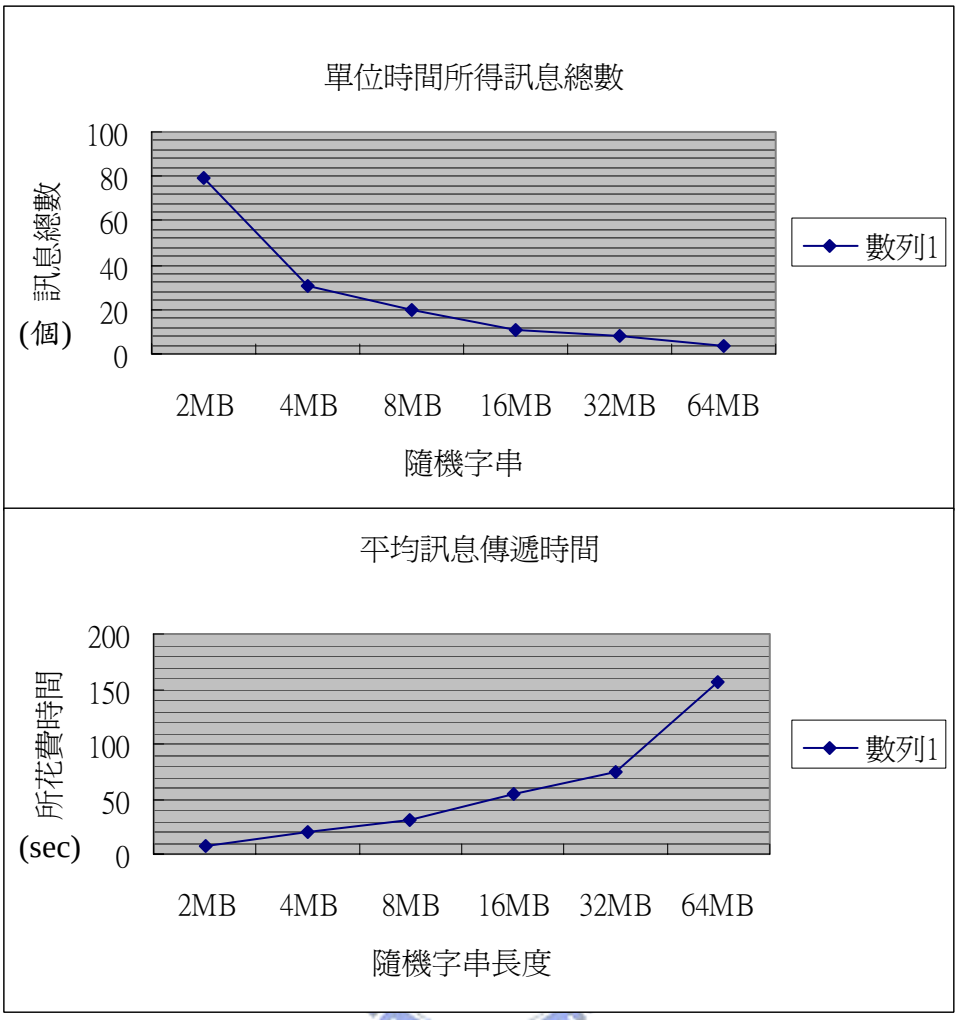
隨機字串長度	比例	總共接收隨機字串長度	失去隨機字串長度
2MB	0.617188	3.24MB	1.24 MB
4MB	0.603708	6.62 MB	2.62 MB
8MB	0.605911	13.2 MB	5.2 MB
16MB	0.695348	23.01 MB	7.01 MB
32MB	0.747861	42.78 MB	10.78 MB
64MB	0.692986	92.35 MB	28.35 MB

表格 3

在表格 3 中所要表達的是，我們實作方法的效能。因為我們的方法主要是兩個使用者之間先接收隨機字串，然後在經由取交集的動作來使雙方擁有相同的隨機字串。經由模擬實驗的結果在總共聽取 3.24MB 的隨機字串時，才會得到 2MB 相同的隨機字串，這樣的比例是 0.617，而我們多聽取沒用的隨機字串是 1.24MB。如果比例越高的話那我們模擬的效能就會越好。

我們將表格 2 用折線圖的方式表現如圖表 13。上方的圖為，在花費時間約十分鐘的情況下，接受相同的隨機字串長度為 2MB~64MB 時，所能得到的 one-time pad 個數。而在下方的圖為每個 one-time pad 的平均傳遞的時間。很清楚的可以看到，隨著隨機字串越長所能得到的個數就越少。但是如果我們都以 2MB 的長度為單位，雖然可以得到很好的效能，攻擊者卻很容易的可以儲存住整個隨機字串的資訊，所以並非以 2MB 為單位就是好的，必須要考量攻擊者的能力在何種程度。在實作的考量上攻擊者並不一定擁有無限的計算能力，對於我們所設計的架構，也可加上一些時間的因素進去。如果攻擊者無法很快的解出密文，但又持續接收隨機字串的訊息，當他的儲存空間達到飽和後。若仍沒有解密出任何密文而又想繼續聽取隨機字串，那麼勢必要將之前所儲存的資訊刪除掉，因此會造成一些密文變成是永久性安全的。

我們所提出的架構中，在初始金鑰的分享是利用公開金鑰加密系統來完成的，所以若攻擊者是無限計算能力時，這樣的共享金鑰方式就不安全。但現實生活中攻擊者是被限制的，所以若攻擊者在解密得到初始金鑰所花費的時間，若比我們傳送訊息的時間要來的久，那這次的傳送就還是安全的。當然我們也可以每傳送一個訊息就變換一把初始金鑰，使攻擊者更難攻擊我們的系統，但這樣做也提高了系統的負載程度。



圖表 13 效能分析

所使用到的相關技術.

- **多點傳播(multicast).** 我們使用多點傳播的方式來模擬一個可以廣播大量隨機字串的伺服器，多點傳播所送出的封包應該是只有對該封包有興趣者才會接收。換句話說也就是只有那些參與多點傳播群組的主機才會接受封包。這也是多點傳播比廣播好的地方，就是它可以降低那些對多點傳送封包沒有興趣之主機的負荷。此外廣播一般都只限制在 LAN(local area network)上，而多點傳播可以應用到 LAN 上或是跨 WAN(wide area network)。但不幸的是在 LAN 上做多點傳播是很容易的，在 WAN 就牽涉到繞送(routing)的問題，所以如果要在 WAN 上做到多點傳播，必須使會用到的路由器(router)都有多

點傳播繞送協定(multicast routing protocol)，使其能夠幫忙傳送多點傳播封包。

第六章 總結

在此論文中我們研究了 Dziembowski-Maurer 的方法並將其安全性做了一個延伸使其能夠抵擋動態攻擊。並且使這樣系統中的初始金鑰，能夠重覆使用而系統仍是安全的，之後提出一個擁有可以捏造性質的加密系統，其主要概念是利用對密文做訊息驗證，在此我們所使用的訊息驗驗方法是 *HMAC*。我們也說明了這個不可捏造的加密系統是可以抵擋 CCA 的。最後我們也設計了一個基於儲存限制的訊息傳送系統架構，所使用的加密方法是我們在第四章所提到的演算法二，並且在一般的乙太網路上模擬這樣的設計。而我們利用特別設計的資料結構與協定，使得在我們所設計的系統裡不需要做時間同步的機制，然後分析所得到的效能。

未來的研究方向，我們所提出的架構在傳送共享金鑰時，還是使用現有的公開金鑰系統，利用加解密方式來共享初始金鑰。因此在共享金鑰這一部分，就必須限制攻擊者的計算能力了。若能夠利用一個基於儲存限制的金鑰共享協定(key agreement protocol)，或是利用量子計算上的共享金鑰方式，那就能夠使得系統的假設是完全不限制攻擊者的計算能力。

參考文獻

[1] Y. Aumann and M. O. Rabin. Information Theoretically Secure Communication in the Limited Storage Space Model. In *Advances in Cryptology – Crypto'99*, Lecture Notes in Computer Science, vol. 1666, pages 65-79. 1999.

[2] Y. Aumann, Y. Z. Ding, and M. O. Rabin. Everlasting Security in the Bounded Storage Model. In *IEEE Transactions on Information Theory*, vol.48, no.6, pages 1668-1680, 2000

[3] M. Bellare, R. Canetti and H. Krawczyk. Keying Hash Functions for Message Authentication. In *Advances in Cryptology - Crypto'96*, Lecture Notes in Computer Science, vol. 1109, pages 1-15, 1996.

[4] C. Cachin and U. Maurer. Unconditional security against memory bounded adversaries. In *Advances in Cryptology – Crypto'97*, Lecture Notes in Computer Science, vol. 1294, pages 292-306, 1997.

[5] Y. Z. Ding. Oblivious transfer in the bounded storage model. In *Advances in Cryptology – Crypto'01*, Lecture Notes in Computer Science, vol. 2139, pages 155-170, 2001.

[6] Y. Z. Ding. *Provably Everlasting Security in the Bounded Storage Model*. PhD thesis, Harvard University, 2001. available at http://www.deas.harvard.edu/~zong.

[7] D. Dolev, C. Dwork, and M. Naor. Non-malleable Cryptography. *SIAM J. Comp.*, vol. 30, no. 2, pages 391-437, 2000.

[8] S. Dziembowski and U. Maurer. Tight security proofs for the bounded-storage model. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 341-350, 2002.

[9] Y. Z. Ding and M. O. Rabin. Hyper-encryption and everlasting security. In *Proceedings of 19th Annual Symposium on Theoretical Aspects of Computer Science*, pages 1-26, 2002.

[10] C. J. Lu. Hyper-encryption against Space-Bounded Adversaries from On-Line Strong Extractors. In Advances in *Cryptology – Crypto'02*, Lecture Notes in Computer Science, vol.2442, pages 257-271, 2002.

[11] C. S Laih, L. Harn and C. C. Chang. *Contemporary Cryptography and It's Applications*. 旗標出版, 2003.

[12] U. Maurer. Conditional-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology*, vol. 5, no. 1, pages 53-66, 1992.

[13] U. Maurer. Secret key agreement by public discussion from common information. *IEEE Transactions on Information Theory*, vol.39, pages 733-742, 1993.

[14] U. Maurer. Information-theoretically secure secret-key agreement by NOT authenticated public discussion. In *Advances in Cryptology – Eurocrypt'97*, Lecture Notes in Computer Science, vol.1233, pages 1209-225, 1997.

[15] A. J. Menezes, P. C. van Oorschot and S. A. Vanstone. *HANDBOOK OF APPLIED CTYPROGRAPHY*. CRC Press Inc. 1996.

[16] U. Maurer and S. Wolf. Information-Theoretic Key Agreement: From Weak to Strong Secrecy for Free. In *Advances in Cryptology – Eurocrypt'00*, Lecture Notes in Computer Science, vol.1807, pages 351-368, 2000.