國立交通大學

資訊科學系

碩 士 論 文

於隨機稀疏矩陣架構下的

分散式金鑰產生機制

Random Sparse Matrix Distributed Key Generation Protocol

研 究 生：歐政儒

指導教授：曾文貴　教授

中 華 民 國　九 十 四　年 六 月

於隨機稀疏矩陣架構下的分散式金鑰產生機制

學生：歐 政 儒　　指導教授： 曾 文 貴 博士

國立交通大學資訊科學研究所

摘 要

分散式金鑰產生(Distributed Key Generation)在門檻式密碼系統(Threshold Cryptosystem)中扮演很重要的角色，而關於建立在離散對數模式下的分散式金鑰產生機制之相關研究已行之有年，由於它所帶來實用的性質因素，而使此議題廣泛且熱切地探討著。儘管這些年來，研究學者們分別提出為數不少的各種架構，但總是在效率上遇到難以突破的瓶頸，尤其是當我們將分散式金鑰產生機制著眼於有著大規模成員數量的系統上時，在成員們為了達到“可驗證之秘密分享(VSS)“之要求，而所需相互傳送的通訊量是很可觀的；因此，在此實際的情形下，各方所提出架構之效能皆不是很出色。

因此，為克服上述的瓶頸，我們提出一個架構於隨機稀疏矩陣的分散式金鑰產生機制，並提出在此架構下有非常高的成功機率以及完整的安全性證明；此外，我們所提的架構亦符合 R.Gennaro 等人於[GJKR99]探討分散式金鑰產生時所提出更嚴謹的安全性定義。

關鍵詞：**分散式金鑰產生、門檻式密碼系統、離散對數、可驗證之秘密分享、隨機稀疏矩陣**

Random Sparse Matrix Distributed Key Generation Protocol

Student：Jheng-Ru Ou　　Advisor：Dr. Wen-Guey Tzeng

Institute of Computer and Information Science

National Chiao Tung University

Abstract

Distributed key generation (DKG) plays a very important role in the threshold cryptosystem and the interrelated researches in the distributed key generation protocol for discrete-log based cryptosystem have been lasted for several years. Due to their realistic factors in all kinds of applications, the discussions for these interrelated research topics are always hot and extensive.

In spite of there are so many distributed key generation protocols proposed by scholars, the performances of these DKG protocols are still not very efficient. Especially when we implement these DKG protocols for the need of achieving “Verifiable Secret Sharing” in some large scale system, the necessary sharing messages sent between members will be very huge. Therefore, the communication load remains so heavy that there are still hard to conquer the bottleneck in efficiency.

To this end, we will propose a new distributed key generation protocol which is constructed on “random sparse matrix”. Then we will also put forward the complete security proof and we claim that there would be very high successful probability in our random sparse matrix DKG. Furthermore, our random sparse matrix DKG also corresponds with the stricter security definition proposed by R.Gennaro etc. in [GJKR99] for

distributed key generation.

Keywords: Distributed Key Generation, Random Sparse Matrix Threshold Cryptosystem, Verifiable Secret Sharing, Discrete-Log.

誌謝

首先感謝我的指導老師曾文貴教授，在我碩士班的學習過程中，帶領我深入密碼學的領域，老師認真積極的教學態度，使我受益良多。另外，我要感謝口試委員，交大資工陳榮傑老師、蔡錫鈞老師與清大資工孫宏民老師，在論文上給我許多建議與指導，讓我的論文更加完善。除此之外，我要感謝實驗室同學，建宇、吳祐與孝盈的幫忙，實驗室學長，成康的指導，以及實驗室學弟們在精神方面的鼓勵。

最後，我要感謝我的家人及女友，不論在精神或物質上都給我極大的支持，讓我能無後顧之憂的情況下，順利完成學業。

在此，謹以此文獻給所有我想要感謝的人。

目次

第一章 引言

本篇所提出的以隨機稀疏矩陣爲架構的分散式金鑰產生機制是建立在離散對數(discrete-log)的安全基礎下，且此系統能在進行公開/私密金鑰產生之總成員人數爲 n 時，只需有超過某門檻數量 t 的成員同意之下，就能重建解密或簽章時所需的私密金鑰 (也就是所謂的“門檻式密碼系統”)；此外，更值得一提的是，在我們所提的架構下，對於大規模使用者所參與的分散式金鑰產生機制之運作，在效率上有很大的突破，也更進一步地改進了 J. Canny 等人爲了解決於大規模成員狀態下，而在[CS04]所提出的分散式金鑰架構的一些缺失。

1.1 研究動機

由於資訊技術的日新月異、網際網路無遠弗屆的應用魅力，它所帶來的通訊便利性也改變了許許多多原來必須面對面才能處理的溝通互動型態；當然，在密碼學領域中的許多研究學者也開始去思考—如何將過往所提出的一些密碼學架構應用於我們現在生活所及的網路世界中？當我們所面臨的使用者不再是僅僅幾十人或幾百人，而是透過網路所連接的上萬人以上的使用者時，那我們該如何因應？當我們將架構置於網路上時，所面臨的不再單單是密碼學理方面的考量，亦須考慮到許多網路技術所產生的特有因素影響時，例如：叢集錯誤(burst error)、阻斷服務(DoS)…等等，那我們要如何找出解決之道呢？

本篇所將探討的“分散式金鑰之產生”也面臨此問題！因此，我們將以能符合大規模成員參與情況之下，在發生叢集錯誤或面臨惡意攻擊者的阻斷服務干擾時有較不受影響的絕佳表現，且尚能維持嚴謹的安全性前提之下，提出一個具有實際應用潛力的分散式金鑰產生機制。

1.2 研究重點

由上一小節，我們可以得知：當我們希望把一個在學理上有出色的表現及在實用上有著不錯的應用潛力之理論架構發展到現實生活中的運用上時，所需要克

服的問題一定遠遠超出原本在學理上的理論；正如先前所述，雖然關於分散式金鑰產生機制的研究極早就備受關注，且相關的原理及技術爲數不少，但在實際運用於大規模成員數量時的效率上，總是沒有很出色的架構提出很有效的解決方法；直到 J. Canny 及 S. Sorkin 在[CS04]發表出一個以對角帶狀稀疏矩陣架構下的分散式金鑰產生機制，將互相所需的通訊量減低很多，因此在效能上才有了很大的突破。

儘管如此，J. Canny 等人之架構在分送秘密分享訊息時，對於網路上的叢集錯誤(burst error)之抵抗能力非常不理想；另外，由於在演算法中所提供的一些初始公開訊息上有些瑕疵，因此，在對於有惡意攻擊者參與的情形之下，容易發生誠實的成員暴露出與秘密分享之分送的相對應身分，進而讓惡意攻擊者能鎖定其攻擊對象，再以假扮或阻斷服務 DoS (Deny of Service)等進行網路相關之攻擊；除此之外，此篇在一些初始的參數設定上有著許多限制。

因此在本篇論文中，便針對我們在探訪 J. Canny 等人在[CS04]所提的分散式金鑰產生架構時，所發現的一些尚能改進之處，提出絕佳的解決之道；我們將利用隨機稀疏矩陣的特性，提出一個在隨機稀疏矩陣架構下的分散式金鑰產生機制，它除了保有[CS04]的效率及[GJKR99]的嚴謹安全性之外，對於網路上傳播訊息時的叢集錯誤影響以及對惡意攻擊者的抵擋，有較爲出色的抵抗能力，且對於一些參數設定上，與[CS04]相較之下，有著更大的彈性調整空間。

1.3 各章節介紹

在以下的章節中，第二章首先針對過往對於分散式金鑰產生此議題所提出的一些較重要之架構做概略性介紹，並闡述其各優缺點及相關演進。在第三章的 3.1 小節到 3.5 小節中，將引進我們所使用的隨機稀疏矩陣之分散式金鑰產生，在成功機率的相關證明時，所將牽涉到一些重要且關鍵的圖論基本技術；並於 3.6 小節及 3.7 小節分別提供在分散式金鑰產生協定中，極重要的相關安全定義

以及篇中將會運用到的凡德蒙地矩陣。而在進入我們的主題之前，我們先在第四章的 4.1 節探討利用矩陣的架構來詮釋傳統的多項式方式的分散式金鑰產生協定之可行性；於 4.2 節來探討傳統的分散式金鑰產生協定之不足處，以及由 J.Canny 等人於[CS04]所提出之目前唯一的矩陣架構。第五章中，5.1 小節及 5.2 小節將詳細說明本篇所倚賴的主要演算法以及此演算法的主要構想說明；於 5.3 小節中，以圖例的方式來闡述我們所提出的隨機稀疏矩陣之構造；5.4 小節將針對我們的隨機稀疏矩陣之分散式金鑰產生架構做更深入的特性分析；而 5.5 小節將說明如何把第三章所介紹的 Random Bipartite Graph 以及 perfect matching 性質對應到我們的架構中，然後藉此來探討我們所提架構的成功機率；進而於 5.6 小節裡探究我們所提出架構的金鑰產生之正確性；接著在 5.7 小節中，我們以模擬(simulation)的方式來分析我們的安全性；然後 5.8 小節將針對本篇所提出的新架構，與之前 J. Canny 等人所提出的相關架構比較之下，探討本篇之主要貢獻；由於爲了實際驗證我們所提出的架構之正確性及其實用性，我們針對[CS04]篇中之架構及本篇所提之架構，實際利用撰寫程式的方式去比較兩者之間的差異及其優劣性，於是我們在第六章中呈現出我們的實驗結果，並提出其詳細的比較分析及探討。最後，我們於第七章對我們所提出的架構做個總結。

第二章 相關研究

在進入本篇所提出的隨機稀疏矩陣架構下之分散式金鑰產生之前，我們先透過以下各小節來對於分散式金鑰產生協定的主要發展做一個概略性的簡介，並進一步地去探討、分析這些相關架構的優缺點，藉以說明如此演進的緣由。

2.1 Shamir 提出的秘密分享協定

秘密分享協定(Secret Sharing)能讓秘密(secret)不單單只由一個人來掌控。一個較早且較有名的秘密分享協定是由 Shamir 在[Sha79]所提出的；此協定能把一個秘密分享給 n 個成員，然後在此 n 之內的任何 t 個成員就能重建出這秘密，這個協定之概略架構如下：

一個受信任的處理者(trusted dealer)利用秘密s來建構出一個有著t次方的多項式 $f(x)=a_tx^t+a_{t-1}x^{t-1}+\ldots+a_1x^1+a_0x^0$，另外，讓$a_0=s$以及$f(0)=s$，然後多項式中的係數是從$Z_q$中隨機挑選的，在此的q是一個大於s的質數。

接下來這個處理者將此多項式上的 n 個不同點之值(除了 0 點之外)分給每個成員；因此，只要有任意 t 個成員將他們所拿到的秘密分享收集在一起，透過 Lagrange 內插法的方式即可解出此 t 次方的多項式，進而重建出這個秘密(secret)了。

關於秘密分享協定的優點主要有：

(1) 在重建步驟的耗費時間上只有多項式時間(polynomial time)。

(2) 資訊理論上的安全不架構於任何假設上，任何少於 t 個成員接無法得知秘密的內容是什麼。

(3) 受信任的處理者能依需求，有彈性地再加入新的成員。

但是此協定有以下的主要缺失：

(1) 在秘密分享的過程中，除了所有成員之外，還需要額外一個受信任的處理者參與。

(2) 每個成員不能進一步地去驗證所拿到的秘密分享之正確性。

2.2 Feldman 提出的可驗證式秘密分享協定

有鑑於 Shamir 提出的秘密分享架構之上述缺失，Feldman 在 1987 年所提出的“可驗證秘密分享(Verifiable Secret Sharing)”協定(於[Fel87])，這主要是根據原先的秘密分享架構加入了一個可公開讓大家驗證的機制；這協定可算是日後“分散式金鑰產生(Distributed Key Generation)”協定蓬勃發展的重要基石，而關於可驗證秘密分享協定，內容大致如下所示：

在此架構中，依然是使用一個次方數爲t的函式$f(x)=a_tx^t+a_{t-1}x^{t-1}+\ldots+a_1x^1+a_0x^0$且設定$a_0=s$及$f(0)=s$；此外，還需要一個產生子(generator)$g\in G_q$給驗證函式所用，然後可信任的處理者除了將$f(i)=s_i$的值，秘密分送給每個相對應i的成員$P_i$之外，亦將 $g^{a_0}, g^{a_1},..., g^{a_t}$ 廣播出去，然後，每個成員P_i就可以透過下述的驗證函式 $g^{s_i} \equiv \prod_{k=0}^{t} (g^{a_i})^{i^k} \pmod p$來驗證看看自己所拿到的分享值是否正確，若驗證不通過，即代表所拿到的分享值是不被接受的。

雖然 Feldman 的可驗證式秘密分享協定可以克服 Shamir 秘密分享協定所不能被公開地驗證之缺失，但 Feldman 的 VSS 依然有以下待改進之處：

(1) 此協定只建構於計算複雜度的安全架構下，因此，此協定並不能保證在相互通訊的狀況之下，不會洩漏任何資訊。

(2) 在此協定中必須要有一個誠實、可信任的處理者，且他(她)知道這秘密。

2.3 Pedersen 提出的可驗證式秘密分享協定

Pedersen 爲了克服 Feldman 的可驗證式秘密分享協定在安全性上的不足處，而利用承諾(commitment)的構想，於[Ped91a]篇中提出一個符合不洩漏任何資訊的安全 VSS 架構：

可信任處理者在Z_q底下，任意選擇兩個次方數爲t的隨機多項式：$f(x)= a_tx^t+a_{t-1}x^{t-1}+\ldots+a_1x^1+a_0x^0$及$g(x)= b_tx^t+b_{t-1}x^{t-1}+\ldots+b_1x^1+b_0x^0$，且設$f(0)=s$與

a_0=s；來把秘密分享s_i=f(i)及s_i'=g(i)透過安全的方式給相對應索引i的成員，並且將 $g^{a_0}h^{b_1}, g^{a_1}h^{b_2}, ..., g^{a_t}h^{b_t}$ 廣播出去，然後，每個成員就透過驗證式子 $g^{s_i}h^{s_i'} \equiv \prod_{k=0}^{t}(g^{a_i}h^{b_i})^{i^k} \pmod p$來檢驗手中所拿到的分享值是否正確。

雖然 Pedersen 克服了 Feldman 所提出之架構在安全性上的缺失，但在此協定中，依舊需要一個誠實、可信任的處理者來參與私密金鑰的選定及分享等過程。

2.4 分散式金鑰產生協定

最原始且最典型的分散式金鑰產生協定(DKG)可視爲是n個由Feldman所提出的可驗證式秘密分享架構平行發生所衍生出來的機制，也就是每個成員P_i都可以是處理者，然後去分享給其他每個成員他自己所隨機選的秘密z_i，而秘密金鑰就是所有z_i的總合；也由於是由Feldman的可驗證式秘密分享所衍生，因此也保有 $y_i = g^{z_i}$ 的特性，分散式金鑰產生協定藉由此特性就可以透過y_i的相乘而算出公開金鑰y。

2.4.1 Pedersen 提出的分散式金鑰產生協定

Pedersen 爲了屏除在一個門檻式密碼系統中(有 n 個成員參與分享在他們組織中的一把共同私密金鑰，當他們想解出以這把私密金鑰加密過的密文時，必須透過這些參與成員中至少有 t(假設門檻爲 t)個成員一起合作，才能解出)，必須有額外一個誠實且可信任的處理者來參與整個私密金鑰的選定、分享等過程；因而在[Ped91b]中，提出了一個可視爲是第一個分散式金鑰產生協定的構想，在這架構中，私密金鑰的選定、分享及重建的動作完全不須透過某個額外的可信任處理者參與，這個組織中的成員自己就能夠完成這些程序。以下就是此演算法架構的主要內容：

每個成員P_i隨機從Z_q中選取一個次方數爲t的多項式 $f_i(x) = a_{it}x^t + a_{it-1}x^{t-1} + \ldots + a_{i1}x^1 + a_{i0}x^0$，且在此設定$f_i(0) = a_{i0} = z_i$；然後每個成員$P_i$將

$g^{a_{i0}}, g^{a_{i1}}, ..., g^{a_{it}}$ 廣播出去，並且秘密地將$s_{ij}=f_i(j)$傳送給成員P_j；當成員P_i從其他成員手中拿到分享值(share)之後，便透過檢驗式子 $g^{s_{ji}} \equiv \prod_{k=0}^{t} (g^{a_{jk}})^{i^k} \pmod p$來檢驗所拿到的分享值是否正確(上式的j=1~n)；如果對於某個j驗證失敗時，此時的成員P_i便發出一個針對成員P_j的“質疑異議”(complaint)廣播訊息，若持續且累積地接到t個針對成員P_j的質疑異議時，成員P_j便被取消接下來參與重建金鑰過程的權利，並且所有從成員P_j分送出去的秘密分享值也都被標示爲不合法而將不被繼續使用下去。反之，成員P_j便將有異議的s_{ji}值公佈出來，而讓大家來檢驗此值之合法性，若依然不能通過檢驗，該成員P_j亦將視爲不合法的成員。因此，透過上述的檢驗過程，便會產生出一群符合後續重建金鑰過程的合法成員(在此假設此合法成員群之集合爲QUAL)

接下來每個成員P_i便來計算$x_i = \sum_{j \in QUAL} s_{ji} \pmod q$；在此附帶一提，其實秘密值x就是所有屬於QUAL中的z_i值之加總；

然後，所有成員就可以透過式子 $y = \prod_{i \in QUAL} g^{a_{i0}} = g^x \pmod p$來將公開金鑰 y 給算出來。

因此概括的來說，Pedersen 所提出的分散式金鑰產生協定有以下貢獻：

(1) 第一個在秘密分享上不需要額外一個可信任處理者的協定(也就是第一個提出分散式金鑰產生協定之架構)。

(2) 每個參與秘密分享的成員也可以去驗證自己所拿到的分享是否正確。

但是，這個首度由 Pedersen 所提出的分散式金鑰產生協定，仍然有以下的待改進之處：

它的安全性依然是建立在計算複雜度的安全上，並不能保證在成員互相通訊期間不洩漏任何資訊。

2.4.2 Gennaro 等人提出的分散式金鑰產生協定

Gennaro 等人於[GJKR99]提出一個攻擊者可以透過控制一小部分的秘密分享

成員，而做出影響金鑰在產生過程時的值，而達到偏頗影響(bias)的手段，因此凸顯出此架構一些不安全、不完備之處；最後，他們再利用[Ped91a]，也就是我們在 2.3 小節所介紹的 Pedersen 爲了達到不洩漏任何資訊的承諾(commitment)架構之構想，來建構出更安全的分散式金鑰產生協定，主要的演算法如下所示：

(1)產生出秘密值x：

每個成員P_i先以處理者的角色，隨機選取一個z_i值，並透過Pederson-VSS的架構來做以下的計算：

(a)成員Pi從Z_q中隨機選取兩個次方數爲t的多項式$f_i(x)$及$f_i'(x)$，使得：

$f_i(x)= a_{it}x^t+a_{it-1}x^{t-1}+\ldots+a_{i1}x^1+a_{i0}x^0$及$f_i'(x)= b_{it}x^t+b_{it-1}x^{t-1}+\ldots+b_{i1}x^1+b_{i0}x^0$，然後設定$z_i=a_{i0}=f_i(0)$.

(b)然後，成員P_i計算出對於所有j=1~n，$s_{ij}=f_i(j)$以及$s_{ij}'=f_i'(j) \bmod q$；接著，成員P_i將s_{ij}及s_{ij}'秘密地傳送給成員P_j，並對所有k=0~t，將$C_{ik}= g^{a_{ik}} h^{b_{ik}} \bmod p$廣播出去。

(c)每個成員P_j去檢驗從其他成員手中接收到的分享值，檢驗方式爲對於每個i=1~n，驗證以下式子是否成立：

$$g^{s_{ij}} h^{s_{ij}'} = \prod_{k=0}^{t} (C_{ik})^{j^k} \bmod p \qquad \text{(I)}$$

假如驗證失敗，成員P_j便廣播出一個對於成員P_i的“質疑異議”！

(d)每個被提出“質疑異議”的成員P_i將把s_{ij}及s_{ij}'的值公佈出來讓大家來檢驗。

(e)其他的成員們將去驗證成員P_i所廣播出的s_{ij}及s_{ij}'，如果依舊無法通過等式(I)的檢驗，即將此成員P_i標示爲“不合法”！

(2)透過上述檢驗之後，將會建構出一個所有沒被標示爲“不合法”的成員所構成的集合 QUAL，而且設定他們秘密的分享值(the share of secret)爲 $x_i = \sum_{j \in QUAL} s_{ji}$；值得一提的是，秘密值 x 其實就是 $\sum_{i \in QUAL} z_i$；此外，x'就

是$\sum_{j\in QUAL} s'_{ji}$

(3)萃取出 $y=g^x \bmod p$：

每個屬於QUAL集合中的合法成員P_i將利用Feldman的VSS技術將$y_i= g^{z_i} \bmod p$揭露出來：

(a)首先，每個屬於QUAL集合中的合法成員P_i將對於所有k=0~t-1 的$A_{ik}= g^{a_{ik}} \bmod p$廣播出來。

(b)每個成員P_j利用以下的式子來驗證每個A_{ik}是否合法：

$$g^{s_{ij}} = \prod_{k=0}^{t-1} (A_{ik})^{j^k} \bmod p \qquad \text{(II)}$$

如果對於某成員P_i，透過此驗證式子失敗時，成員P_j便廣播出一個對P_i的“質疑異議”，並且附帶s_{ij}及s_{ij}’值給其他成員。

(c)假設至少有一個對於P_i的合法“質疑異議”被提出來，則所有在QUAL的其他成員將會利用Pederson所提出VSS中的重建方法將z_i、$f_i(z)$及A_{ik}等值重建計算出來。

(d)最後，所有在QUAL集合中的成員將y_i設定爲$A_{i0}= g^{z_i}$之後，便可透過$y=\prod_{i\in QUAL} y_i \pmod p$的式子找出y。

對於 Gennaro 等人提出的分散式金鑰產生之貢獻可歸結爲：

(1) 明確指出 Pedersen 提出的分散式金鑰產生協定不能夠在有攻擊者參與的情況下，保證它的安全性；攻擊者可以在重建金鑰的過程中去影響一些輸出值得分布狀況。

(2) 重新檢視以往對於在分散式金鑰產生協定中的安全定義，修正並提出更嚴格的安全定義。

(3) 在初始承諾階段(initial commitment phase)的設計上，利用於[Ped91a] 篇中 Pedersen-VSS 所提出的承諾(commitment)架構來增強安全性。

2.4.3 J.Canny 等人提出的分散式金鑰產生協定

由目前的介紹看來，Gennaro 等人所提出的分散式金鑰產生協定好像已是無可挑剔，也因此在 2004 年之前，所有跟分散式金鑰產生相關的架構幾乎都以 Gennaro 等人於[GJKR99]篇中所提的演算法爲主要探討典型。但是，J. Canny 等人卻指出：由於傳統的分散式金鑰產生協定中，每個成員所需傳送的分享值必須是自己之外的所有其他成員，這情況若實際應用於有著大規模成員參與的系統時，其所需的非常可觀通訊量將是主要影響此分散式金鑰產生系統在效能的表現上非常不理想的主因；因此，他們於[CS04]篇中，提出一個以對角帶狀稀疏矩陣來取代原先以多項式架構的分散式金鑰產生協定，此舉能讓每個成員所需的通訊量大大降低許多；此構想著實在分散式金鑰產生協定中，一個非常大的突破！

由於這是一個非常創新的做法，因此我們將利用一整個章節(於本篇的第四章)來探討其主要的構想及針對其重要的特性做進一步地介紹；此外亦將針對其不足處之探討，而於第五章正式提出我們的隨機稀疏矩陣之分散式金鑰產生架構！

第三章 相關基本技術與原理

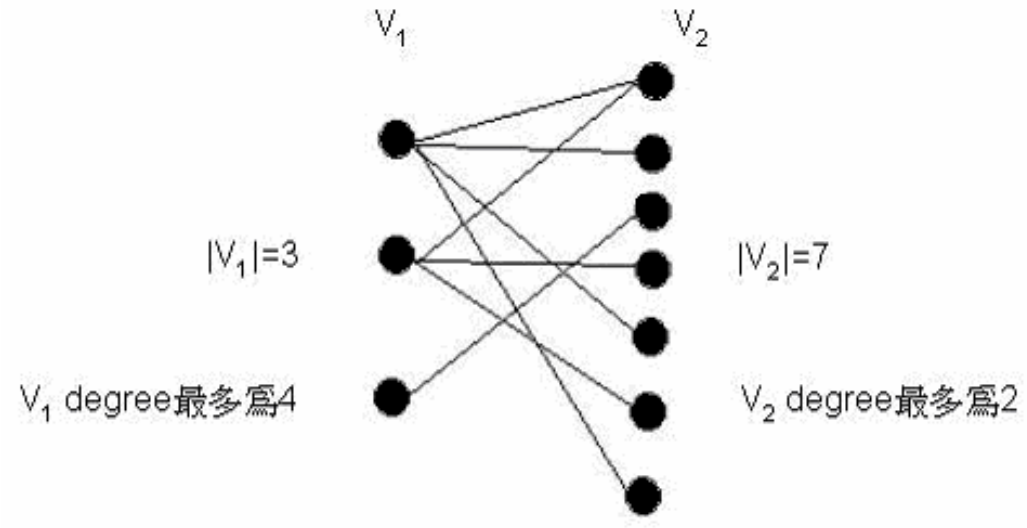
在這一章節中，將會對我們架構中所需運用的基本技術部份，做概略性的介紹及說明。因此，以下各小節將包含 Random Bipartite Graph、Expander Graph、Random Bipartite Graph 與 Expander Graph 之關聯性、Perfect Matching 的廣義及狹義的定義、Hall 於圖論方面的一個探討 bipartite graph 與 perfect matching 之間關聯性的定理、分散式金鑰產生協定的相關安全定義以及線性代數中一個頗重要的凡德蒙地矩陣(Vandermonde Matrix)。

3.1 Random Bipartite Graph 定義

定義 1：

random bipartite graph $G=(V_1,V_2,E)$是指在一個無方向性的graph中，含兩個不重複的端點集合V_1及V_2，edge連接著兩邊集合的端點(如：若edge $(u,v)\in E$,則 $u\in V_1, v\in V_2$或是$u\in V_2, v\in V_1$)；假如$|V_1|=n_1, |V_2|=n_2$且每個在V_1中端點的degree最多爲d_1，每個在V_2中端點的degree最多爲d_2，則此G就稱爲一個有著degree(d_1,d_2)的(n_1,n_2)-bipartite graph。

如下所示，是一個以degree(4,2)的(3,7)-bipartite graph的圖例：



圖一　一個有著degree(4,2)的(3,7)-bipartite graph

3.2 Expander Graph

簡單的來說，一個圖形中，有著少數端點的子集卻能與大量的鄰居端點相連，則我們稱此圖形爲 expander graph；因爲 expander graph 有著許多特殊的性質，因此，不論是在圖論、網路、編碼理論、複雜度運算及 derandomization 等領域中皆佔有很重要的地位。以下，我們會先簡介 expander graph 特性及其與本篇論文所使用之相關特性；然後再針對 expander graph 做較正式的定義。

3.2.1 特性及其應用之概述

由於 expander graph 有許多出色的特性，因此，皆能在很多領域中發現其特殊性質發揮之處；在本篇論文中，我們將利用在網路應用領域中，一個常見的 expander graph 重要特性：在 expander 中，每個在子集中的端點皆有許多相連的鄰居端點存在，也就是說它有著一個很好的擴充性質(good expansion property)；此外，當我們從此 expander graph 中，隨機移除一些相連的接線(edge)時，並不會有顯著的擴充特性減少之影響。

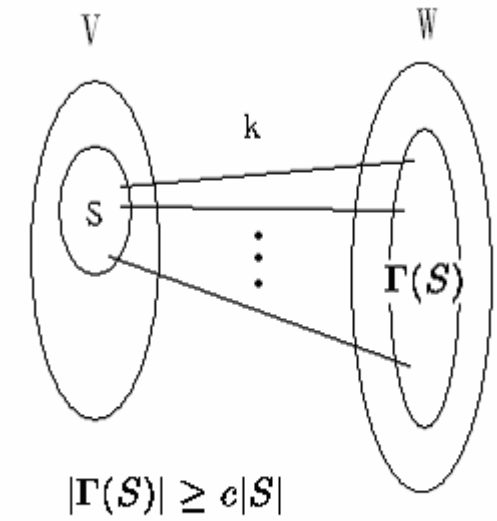
因此，在網路應用領域上，expander graph 的擴充相連之絕佳性質，提供了在建造一個網路時，能保證其端點與端點之間的相連及訊息之路由(routing message)有堅韌(robust)的連接性；所以，藉此就能建造出一個能容錯的網路通訊環境(fault-tolerance network)。

本篇所提的分散式金鑰產生架構，在實際環境情況下，必定不能假設在金鑰分散及重建的過程中，沒有任何惡意攻擊者的參與；因此，我們將於 4.4 小節中，利用上述所提及，應用於網路容錯架構中的 expander graph 特性，來進一步地證明我們的隨機稀疏矩陣之分散式金鑰產生架構在有惡意攻擊者參與的情況下，亦有很高的成功機率達成分散式金鑰產生之目的。

3.2.2 正式定義

定義 2：

在一個 bipartite graph G=(V,W,E)中，假如對於所有基數(cardinality) 最多爲 k 的 S⊆V，|Γ(S)|≥c|S|，在此Γ(S)是 S 之鄰居(neighbor)的一個集合，則稱此 G 爲一個(k,c)-expander



圖二 c-expander

3.3 Random Bipartite Graph 與 Expander Graph

3.3.1 兩者之關聯

一般而言，最常用來判斷一個圖形是否爲一個好的 expander，可以透過計算此圖形的第二 eigenvalue 與第一 eigenvalue 之比值的大小來檢驗，當此比值越小也就表示此圖形越是一個具有良好擴充特性的 expander graph。

可惜的是，目前並未有任何一個比較有效的演算法，可以用來產生出一個具有良好擴充性質的 expander graph；但是，藉由下一小節的定理及其證明可得知，當我們透過隨機產生的 random bipartite graph 會有非常高的機率成爲一個 expander graph。

3.3.2 定理與證明

定理 1：

造出一個在[n]∪[m]的 random bipartite graph G，對於每個在[m]的端點有 d 個在[n]的隨機鄰居；假設 $ce^d \frac{m}{n}(\frac{ck}{n})^{d-1-c} \le \frac{1}{2}$ **，則 G 不是(k,c)-expanding 的機率小於** $2[ce^{c+1}\frac{m}{n}(\frac{2c}{n})^{d-1-c}]^2$

證明：

考慮集合的大小s≤k；一個固定大小爲s的集合有所有在一個特定大小爲cs集合的鄰居之機率最多爲$(cs/n)^{ds}$；則所有任意在[m]中，大小爲s的子集合以及所有任意在[n]中，大小爲cs，其任何大小爲s之集合不爲expand的機率最多爲：

$$C_s^m \cdot C_{cs}^n \cdot (\frac{cs}{n})^{ds} \le (\frac{em}{s})^s (\frac{en}{cs})^{cs} (\frac{cs}{n})^{ds} = (e^{c+1}\frac{cm}{n}(\frac{cs}{n})^{d-1-c})^s$$

接下來，我們令上式最後一個展開式爲p_s，也就是

$$p_s = (e^{c+1}\frac{cm}{n}(\frac{cs}{n})^{d-1-c})^s = (e^{c+1}\frac{cm}{n}(\frac{c}{n})^{d-1-c})^s s^{(d-1-c)s}$$

所以我們可得出：

$$p_s/p_{s-1} = ce^{c+1}\frac{m}{n}(\frac{c}{n})^{d-1-c}(\frac{s^s}{(s-1)^{s-1}})^{d-c-1} \le ce^d\frac{m}{n}(\frac{cs}{n})^{d-1-c}$$

因此，利用 $ce^d \frac{m}{n}(\frac{ck}{n})^{d-1-c} \le \frac{1}{2}$ ，則G不爲expanding的機率最多爲

$\sum_{s=2}^{k} p_s \le p_2 \sum_{s=1}^{k} 2^{1-s} < 2p_2 = 2[ce^{c+1}\frac{m}{n}(\frac{2c}{n})^{d-1-c}]^2$ 即證得！

所以random bipartite graph是expander graph之機率就是：

$$1 - 2[ce^{c+1}\frac{m}{n}(\frac{2c}{n})^{d-1-c}]^2$$

接下來我們就來探討上述所算出的機率$1 - 2[ce^{c+1}\frac{m}{n}(\frac{2c}{n})^{d-1-c}]^2$大約是多少？由於本篇之後的相關證明所運用的是一個 2-expander graph，因此式中的 c 值我們就以 2 的值爲前提之下來討論上述之機率估算；此外，我們的架構中所用到的 m 值最多不大於 n/2，因此，將上述前提及假設值帶入後，我們可以得出以下最低界線的機率：$1 - 2[4e^3(\frac{4}{n})^{d-3}]^2$ ，又由於在本篇論文中，我們是將此 expander

graph 的特性運用於大規模的稀疏矩陣(詳見於 4.6 小節)，因此，上式中的 n 值往往是大於 1000 以上，甚至上萬的數量，且在大規模之結構中，式子中的 d 值也將是二十以上的數值。

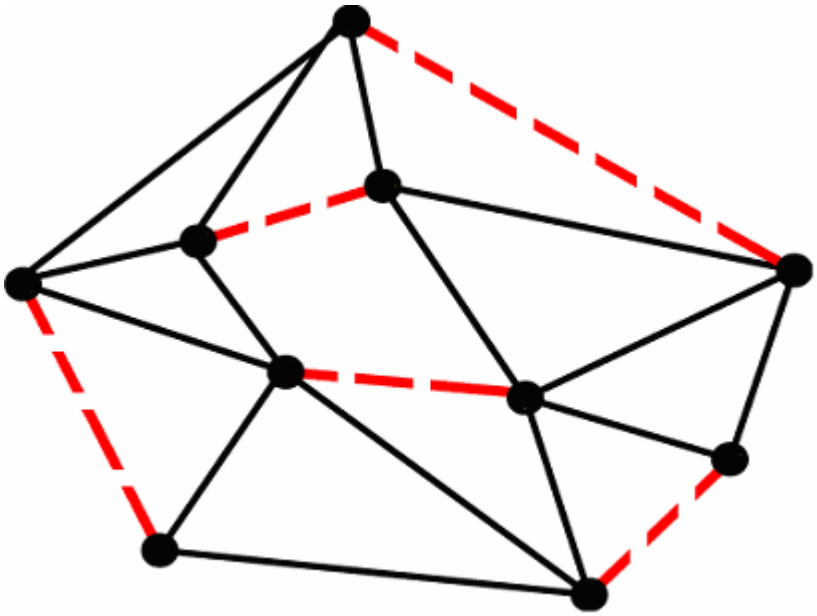
所以，本小節所探討出的機率值非常高而趨近於“1”。

3.4 Perfect Matching

定義 3：

廣義的 Perfect matching 是一種可以涵蓋所有圖上端點的 matching 關係，也就是說，所有圖上的端點都有一個確切的 incident edge。

如下圖所示，各端點接相互透過虛線相連，形成一個 perfect matching：

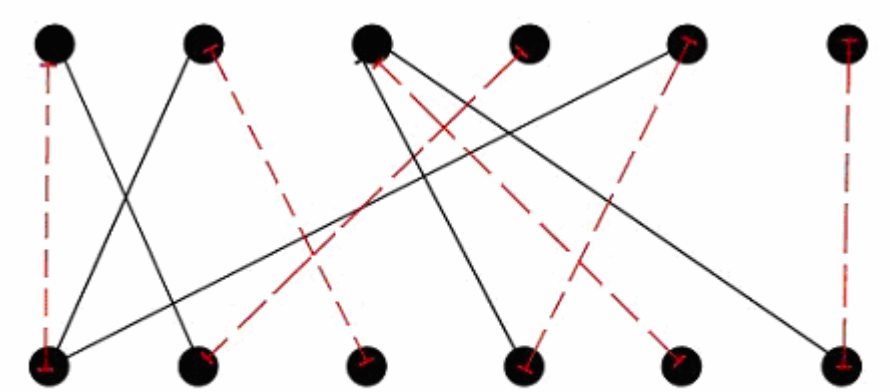


圖三.perfect matching

定義 4：

於 bipartite graph 上的 perfect matching 是指：當上述廣義的 perfect matching 情形是發生於一個 bipartite graph 的兩邊端點集合之間，也就是說，每個左端點集合中的任一個端點會有一個位於右端點集合中的另一端點與之配對。

如下頁圖例所示，兩邊的各端點皆互相透過虛線的連結，形成一個 bipartite perfect matching：



圖四 bipartite perfect matching

3.5 Hall 之 Bipartite 與 Perfect Matching 定理

定理 2：

設圖形G=(V_1,V_2;E)是一個bipartite graph，則此圖形G有一個perfect matching從V_1到V_2，若且爲若，對於所有的包含於V_1的S，有$|\Gamma(S)| \geq |S|$的關係。(I)

而其中bipartite graph的input集合是V_1，其output集合是V_2；$\Gamma(S)$是表示V_2中的集合；且edge集合爲E。

證明：

我們將針對V_1的尺寸大小運用歸納法的討論方式來證明。

當V_1有一個端點時是成立的。

假設我們現在聲稱此定理在V_1小於某m時是成立的。

接下來我們就來考慮以下的兩個可能發生情況：

情況(1) 所有在V_1中適當的子集合S，S不爲空，將之擴充，也就是有$|\Gamma(S)|>|S|$的情形。考慮在V_1中的端點x，讓(x,y)是一個在集合E中的edge。

然後，我們考慮一個有著端點集合$V_1^* = V_1 - \{x\}$及$V_2^* = V_2 - \{y\}$的bipartite graph G*；因爲每個$S \subset V_1$符合(I)，每個V_1^*中的子集合符合(I)，因爲只有單一端點y已經從V_2移除。

因此，運用之前的歸納假設，這個較小的圖形G*有個matching；而針

對此matching，我們將edge(x,y)加入；這樣一來，就有得出一個perfect matching於圖形G中。

情況(2) 存在一個適當的子集合$T \subset V_1$，此T不爲空，且有著$|\Gamma(T)|=|T|$的關係。

我們考慮induced圖形G_1及G_2於端點集合$T \cup \Gamma(T)$，而且尚有以下的關係$V_1 \setminus T \cup V_2 \setminus \Gamma(T)$。

接著運用之前的歸納假設，G_1有一個perfect matching。

(注意，之前的歸納假設不能直接地運用於G_2)

設定$S \subseteq V_1 \setminus T$，然後可導出：

$$\Gamma_{G_2}(S) = \Gamma_G(S \cup T) \setminus \Gamma_G(T) \Rightarrow |\Gamma_{G_2}(S)| \geq |S \cup T| - |T| = |S|$$

而上式中的“≥”是存在的，因爲S∪T符合$|\Gamma_G(S \cup T)| \geq |S \cup T|$，且由於在此的假設是$|\Gamma(T)|=|T|$；因此，圖形$G_2$也符合(I)，而且運用之前的歸納假設，會有一個matching。所以在G_1與G_2的perfect matching之聯集便是一個於圖形G的一個matching。

3.6 分散式金鑰產生協定之相關安全定義

以下所將要介紹的兩個與分散式金鑰產生協定相關的安全定義之要求，是假設p是眾所皆知的一個大質數；q也是一個屬於p-1 因數的大質數；g是一個order爲q且在Z_p中的元素。

而在定義一中的前面三個要求，是一般在檢驗一個以離散對數爲基礎下的分散式金鑰產生協定是否符合安全的三大要求。

由於 Gennaro 等人針對當時所提出的一般分散式金鑰產生協定，設計出一個惡意攻擊者可以經由在[GJKR99]篇中所述的攻擊手法，而在金鑰重建過程中，偏頗金鑰產生的值；也因此，Gennaro 等人也就針對此安全性上的不足處，提出更嚴謹的分散式金鑰產生協定之安全要求，也就是新增了底下的第四項要求；詳細的定義說明將如下頁所述：

定義 5：

一個 t-secure DKG 演算法在假設被攻擊者控制的成員數量是小於 t 的情況下，符合以下要求：

(C1) 所有 t+1 個誠實成員所提供的分享值之集合會定義出相同且唯一的私密金鑰(secret key) x 值。

(C2) 所有誠實的成員會得出相同的公開金鑰(public key) $y=g^x \bmod p$。(而在此的x是上述(C1)所指的那個唯一x值。)

(C3) 在Z_q中，x是隨機均勻分佈(uniform)。

(S1) 攻擊者無法得到任何關於x的資訊，除非它能解$y=g^x \bmod p$。

因爲本篇所提出的分散式金鑰產生協定是一個含有機率性的架構，也就是在本篇論文中所運用的演算法，是容許在有很小的機率發生錯誤之假設前提下；因此，爲了使本篇演算法在安全定義上之分析地更貼切，在下述的安全性定義二中，將把之前所提的定義一，融合我們架構中的機率性質而做些修改，進而提出了以下的安全性定義：

定義 6：

一個機率性(α,β,δ) 門檻式之分散金鑰產生協定(probabilistic (α,β,δ) threshold DKG)演算法，是在假設被攻擊者控制的成員數量是小於α n 的情況下，符合上述的(C2)、(C3)、(S1)以及下面所述的(C1')和(C4')：

(C1') 任何誠實成員的分享值之子集合定義出相同的金鑰 x

(C4') 任何數量至少爲β n 的誠實成員子集合有 1-δ 的機率可以重建金鑰

從上述定義可知，一個 t-secure DKG 演算法就是一個機率性門檻式爲：$(\frac{t}{n},\frac{t+1}{n},0)$的分散式金鑰產生協定之演算法形式。

3.7 凡德蒙地矩陣(Vandermonde Matrix)

在實驗數據的處理上，對於一個以x_i爲輸入而以yi爲輸出的系統而言，常會出現已知n組數據 (x_1, y_1)、(x_2, y_2)、…、(x_n, y_n)，而想知道若輸入爲x_0時，其最可能的輸出值y_0是多少？

(1) 若已知數據有兩組，則可以使用內插法或外插法；

(2) 若已知數據有三組，則可以使用二次曲線來取代線型進行內插或外插法；

(3) 若已知數據有n組，則一個常用的想法是利用n-1 次的多項式作爲分析的工具；這個想法是設想輸入x_i與輸出y_i之間有以下如此的關係：

$c_0+c_1x_i+c_2x_i^2+\ldots+c_{n-1}x_i^{n-1}=y_i$ 且i=1~n

再應用已知的n組數據以求出係數c_i之後，再代入要分析的輸入x_0，而得到要估計的輸出y_0。其中尋找係數c_i是經由解以下聯立方程組而得到的：

$$c_0+c_1x_1+c_2x_1^2+\ldots+c_{n-1}x_1^{n-1}=y_1$$
$$c_0+c_1x_2+c_2x_2^2+\ldots+c_{n-1}x_2^{n-1}=y_2$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$c_0+c_1x_n+c_2x_n^2+\ldots+c_{n-1}x_n^{n-1}=y_n$$

上述聯立方程組可以利用矩陣的概念將之轉換爲以下形式：

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \ldots & x_2^{n-1} \\ \ldots\ldots\ldots\ldots\ldots\ldots & & & & \\ 1 & x_n & x_n^2 & \ldots & x_n^{n-1} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \ldots \\ c_{n-1} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ .. \\ y_n \end{bmatrix}$$

而有著以下形式的矩陣就稱爲凡德蒙地矩陣(Vandermode Matrix)：

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \ldots & x_2^{n-1} \\ \ldots\ldots\ldots\ldots\ldots\ldots & & & & \\ 1 & x_n & x_n^2 & \ldots & x_n^{n-1} \end{bmatrix}$$

第四章 矩陣架構之分散式金鑰產生概述

4.1 傳統 DKG 改變為矩陣構造之可行性

在進入我們所提出的隨機稀疏矩陣之分散式金鑰產生架構之主題前，我們先進一步地來探討爲何以矩陣的方式來架構分散式金鑰產生協定是可行的？也就是本小節將針對之前由 Gennaro 等人於[GJKR99]中所提出之較完備且嚴謹的分散式金鑰產生協定，利用矩陣的觀點來對其架構做技巧性的置換，以引導出我們即將闡述的隨機稀疏矩陣之金鑰產生架構。

簡而言之，我們可以把在 2.4.3 小節中，由 Gennaro 等人提出的最典型且安全的分散式金鑰產生架構中，所需的多項式型態改爲以凡德蒙地矩陣(Vandermonde Matrix 詳見於 3.6 小節)的矩陣型態來置換其演算法中的結構，如下圖所示：

$$
\begin{aligned}
&a_{i0}+a_{i1}x_1+a_{i2}x_1^2+\ldots+a_{im-1}x_1^{m-1}=s_{i1}\\
&a_{i0}+a_{i1}x_2+a_{i2}x_2^2+\ldots+a_{im-1}x_2^{m-1}=s_{i2}\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&a_{i0}+a_{i1}x_n+a_{i2}x_n^2+\ldots+a_{im-1}x_n^{m-1}=s_{in}
\end{aligned}
$$

上述 n 個次方數爲 m-1 的多項式組，可以用下列所示的矩陣方式置換：

$$
\begin{bmatrix} a_{i0} & a_{i1} & \ldots\ldots\ldots & a_{im-1} \end{bmatrix}
\begin{bmatrix}
1 & 1 & \ldots\ldots\ldots\ldots & 1\\
x_1 & x_2 & \ldots\ldots\ldots\ldots & x_n\\
x_1^2 & x_2^2 & \ldots\ldots\ldots\ldots & x_n^2\\
\ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots\\
x_1^{m-1} & x_2^{m-1} & \ldots\ldots\ldots\ldots & x_n^{m-1}
\end{bmatrix}
=
\begin{bmatrix} s_{i1} & s_{i2} & \ldots\ldots\ldots\ldots & s_{in} \end{bmatrix}
$$

假設我們讓中央的矩陣(在此假設爲矩陣E)爲一個有著m列數以及n行數的凡德蒙地矩陣，而在此的m爲分散式金鑰產生系統的安全門檻值；因此，$E_{ij}=j^{i-1}$。每個a_i是一內含有m個元素的隨機行向量，也就是一個內含a_{i0} ~ a_{im-1}的列向量，且向量a_i與矩陣E之相乘也就造出外部秘密s_i，由上圖可知，此s_i是一個含有n個元素的列向量，所以在Gennaro等人於[GJKR99]中所提的演算法中之檢驗群體$|Q_i|=n$，

此等同於在傳統的分散式金鑰產生協定中，於內部秘密及相關係數所構成多項式上的點 1,2,…,n。

而秘密金鑰也就是一個運用於合法成員們之內部秘密總合的式子。每個成員相關於此秘密之分享值就是將手中所有從其他合法成員得來的分享值加總；相同地，每個成員之分享就是在由所有合法成員之多項式加總所得之多項式上的某個點。因爲所有的多項式都是 m-1 次方，$\sum_{i\in V} s_{ij}$ 中的任何 m 個值，可利用內插法來重建出 $\sum_{i\in V} a_i$ ，因此，只要有 m 個合法的分享值，演算法就能成功地建構出公開或秘密金鑰。(而在矩陣的觀點來說，當矩陣 E 的列及行之最大獨立數爲 m 時，也就是秩爲 m 的全秩時，所分享出的分享值就能藉由演算法，成功地重建金鑰。

4.2 J. Canny 等人的分散式金鑰產生協定

4.2.1 傳統的分散式金鑰產生協定之不足處

由於Gennaro等人所提出的分散式金鑰產生協定在安全方面的表現近乎完善，但是J. Canny等人卻發現：在Gennaro等人提出的架構中，當每個成員以中心處理者角色分送秘密分享時的對象必須是除自己之外的所有成員，也就是上一小節中，每個成員所需傳送分享值的檢驗群體$|Q_i|=n$的尺寸太大！這架構若運作於有著大規模成員數量的系統時，在通訊量上的耗費是非常可觀的！

4.2.2 對角帶狀稀疏矩陣之分散式金鑰產生

因此，J. Canny等人便依此實際且實用的需求，於[CS04]提出一個以離散對數(discrete-log)爲基礎的對角帶狀稀疏矩陣分散式金鑰產生協定，藉由此架構，能讓每個成員在扮演中心處理者分送秘密分享時的通訊量，把每個成員所需傳送給全部n個成員之通訊量降到一個極低的固定值，也因此讓分散式金鑰產生在實際大規模的運用上更有效率且更有實用性。所以，J. Canny等人便針對我們於 4.1

小節中所述的凡德蒙地矩陣之構造作一番檢視後，利用矩陣相成的特性，將原先的凡德蒙地矩陣(也就是 4.1 節中的矩陣E)置換爲他們所提的“對角帶狀稀疏矩陣”的構造，運用稀疏矩陣與a_i向量相乘後，所得出的s_i向量中所含的非零值之元素數量會大大減低，進而讓每個成員所需傳送的分享值之通訊量盡可能地降至最低，此架構之主要想法可由下圖所示得知：

由原先的凡德蒙地矩陣架構：

$$\begin{bmatrix} a_{i0} & a_{i1} & \cdots & a_{im-1} \end{bmatrix} \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \cdots & \cdots & \cdots & \cdots \\ x_1^{m-1} & x_2^{m-1} & \cdots & x_n^{m-1} \end{bmatrix} = \begin{bmatrix} s_{i1} & s_{i2} & \cdots & s_{in} \end{bmatrix}$$

改爲以下的對角帶狀稀疏矩陣：

$$\begin{bmatrix} a_{i0} & a_{i1} & \cdots & a_{im-1} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & & & \\ & 1 & 1 & 1 & 1 & 1 & & \\ & & \cdots & \cdots & \cdots & & & \\ & & & & 1 & 1 & 1 & 1 & 1 \\ & & & & & 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} s_{i1} & s_{i2} & \cdots & s_{in} \end{bmatrix}$$

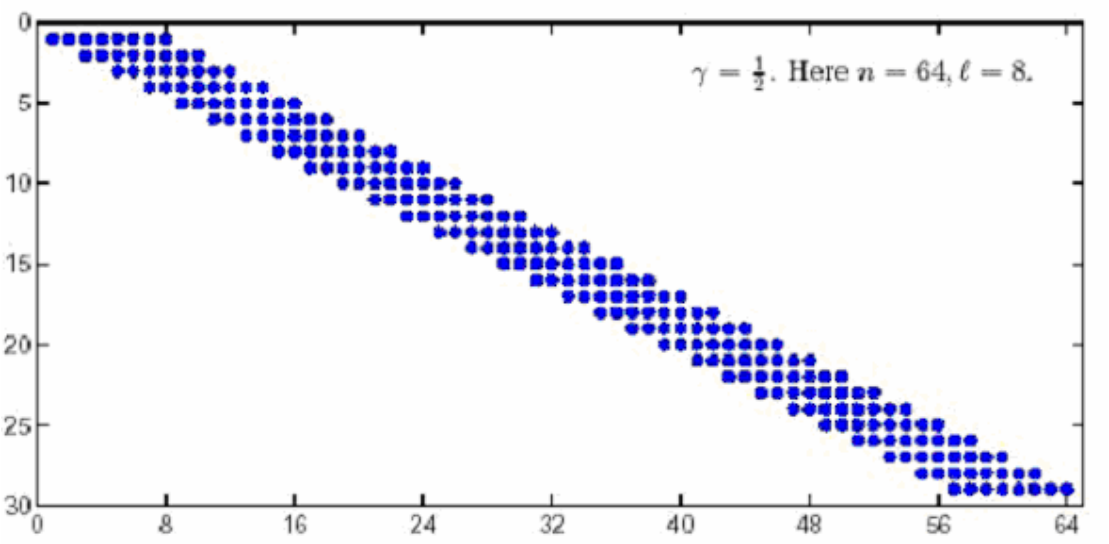
u continue non-zero

上圖中的a_i是一個內含只有u個連續出現的非零值之元素的列向量，而在此爲了方便審視其對角帶狀稀疏矩陣的構造，其內含的非零值以 1 來表示。

有了上述的對角帶狀稀疏矩陣之分散式金鑰產生協定的概念後，現在我們就針對此方法做個叫詳細的剖析：

以下是以一個行數 n 爲 64，列數 m 爲 29 的矩陣爲例；且此矩陣是一個以下述規則所產生的帶狀稀疏矩陣：在這個矩陣的每一列中只有 l 個非零且連續出現的值(在此的 l 爲 8，也就是圖中所標示的著色點)，其他矩陣中大部分的元素都是爲

零；然後，上一列跟下一列的l個非零值之起始位置會往右位移γ^{-1}行；因此，這個矩陣就會呈現出一個附有對角帶狀的稀疏矩陣，且此矩陣的行數與列數之間的關係會因l及γ^{-1}的值變動而有所不同，也就是有$m \approx \gamma(n-l)+1$的關係存在。



圖五 對角帶狀稀疏矩陣

有了上述的對角帶狀稀疏矩陣之後，接下來便利用以下由 Gennaro 等人提出之演算法，針對帶狀稀疏矩陣之特性做細部的些微修改即可。

J. Canny 等人的主要架構概念如下所述：

將原本在 Gennaro 等人所提出的分散式金鑰產生演算法中，每個成員在扮演中心處理者角色時所隨機選的兩個次方數爲 m 的多項式，改以兩個隨機選定且尺寸爲 m x 1 之列向量(此列向量之其中只含連續 u 個非零值之元素)分別與上述的帶狀稀疏矩陣 E 相乘之值來置換；因此，當列向量與稀疏矩陣相乘運算後，只有少數的值是非零，再將非零值的相對應成員設爲所需的秘密分享分送對象，如此一來，所需分送的訊息通訊量便大大減低很多，此後，將 Gennaro 等人所提的分散式金鑰產生協定之演算法中的多項式概念改以向量及矩陣的相對應方式，再依循其演算法之步驟即可發展出一個以對角帶狀稀疏矩陣之分散式金鑰產生架構。

此架構所呈現的主要貢獻大致有以下四點：

(1) 利用對角帶狀稀疏矩陣的概念提出一個新的分散式金鑰產生架構，在此新架構中的分送秘密分享之對象不需要是全部的成員，因此大大減少了通訊量上的負擔。

(2) 利用隨機漫步(Random Walk)之特性來證明此新架構可以抵擋有總數爲 $\frac{1}{2}-\varepsilon$ 的惡意成員參與之下的干擾攻擊。

(3) 此架構符合 Gennaro 等人在[GJKR99]篇中，對於分散式金鑰產生協定所訂出的更嚴謹之安全定義。

(4) 在效率上，跟之前 Gennaro 等人提出的分散式金鑰產生架構比較起來更有效率，且在大規模成員參與之系統上更爲實用。

但是，經由深入的探究之後，我們發現此篇由 J. Canny 等人提出的架構依舊有許多待改進之處，我們將在下一章針對他們架構的缺失做進一步的剖析，進而引導出我們所提出的更完善架構。

第五章 隨機稀疏矩陣之分散式金鑰產生

5.1 主要演算法

藉由第二章與第四章的相關秘密分享及傳統或矩陣架構分散式金鑰產生協定之概述之後，對於 DKG 之主要原理及其構造也就應該就會有更深一層的認識。因此，接下來便來介紹本篇的隨機稀疏矩陣之分散式金鑰產生所套用的主要演算法，其主要架構是由 2.4.2 節中所提到的 Gennaro 等人於[GJKR99]篇中演算法，針對矩陣的形式及其特性作些微的修改，演算法內容如下所示：

(1)以一個分配處理階段(dealing phase)爲開始，以致於所有的成員都知道E、v、g、h、p、q，在此的p是一個大質數，而q是一個被p-1 整除的大質數，g及h是在Z_p中order爲q的元素，E是一個m x n在Z_q的矩陣，而v是一個在Z_q中含有m個元素的向量。

(2)產生出 x：

(a)每個成員i選出兩個列向量，a_i及$a_i' \in Z_q^m$

(b)然後，成員i計算出$s_i=a_iE$以及$s_i'=a_i'E$，定義出檢驗群體(checking group)爲$Q_i=\{j|s_{ij} \neq 0 \vee s_{ij}' \neq 0\}$，接著，成員i將$s_{ij}$及$s_{ij}'$的第j個元素傳送給屬於$Q_i$中的成員j，並且將$C_i= g^{a_i} h^{a_i'} \bmod p$廣播出去給所有$j \in Q_i$

(c)每個成員j去檢驗從其他成員接收到的分享值，檢驗方式爲對於每個成員$j \in Q_i$，驗證以下式子是否成立：

$$g^{s_{ij}} h^{s_{ij}'} = \prod_{k=1}^{m} (C_{ik})^{E_{kj}} \bmod p \qquad \text{(Eq.1)}$$

假如驗證失敗，成員j便廣播出一個對於成員i的“質疑異議”給Q_i

(d)每個被提出“質疑異議”的成員i將把s_{ij}及s_{ij}'廣播出給Q_i

(e)其他在Q_i中的成員們將去驗證成員i所廣播出的s_{ij}及s_{ij}'，如果依舊無法通過等式(I)的檢驗，即將此成員i標示爲“不合法”！

(3)每個成員i建構出一個所有沒被標示爲“不合法”的成員所構成的集合V，而且設定他們秘密之分享(share of secret)爲 $x_i = \sum_{j \in V} s_{ji}$ ；值得一提的是，x_i其實就是向量 $(\sum_{j \in V} a_j)E$ 的第i個元素；此外，私密金鑰定義爲x= $(\sum_{i \in V} a_i)v$

(4)顯露出(reveal) $y=g^x \bmod p$：

(a)每個屬於V的合法成員i將向量$A_i = g^{a_i} \bmod p$廣播給Q_i

(b)每個成員$j \in Q_i$利用以下的式子來驗證每個A_i是否合法：

$$g^{s_{ij}} = \prod_{k=0}^{m} (A_{ik})^{E_{kj}} \bmod p \qquad \text{(Eq.2)}$$

如果對於i此驗證式子失敗時，成員j廣播出一個對i的“質疑異議”附帶s_{ij}及s_{ij}’給Q_i

(c)假設至少有一個對於i的合法“質疑異議”被提出來，則所有在Q_i的成員將會公開地利用解$s_i=a_iE$的線性等式來重建a_i

(d)每個在Q_i的誠實成員知道成員i是否合法，如果是，則A_i的值是正確的；而爲了找y，可詢問每個在Q_i的成員，找到集合V，然後，透過式子 $y=\prod_{i \in V} \prod_j A_{ij}^{v_j}$ 即可找出y

5.2 演算法之流程說明

5.2.1 秘密之分享

參照 5.1 小節的演算法內容，我們可以知道矩陣E是一個含有m列及n行的矩陣，而其中的n也就是代表著所有在分散式金鑰產生系統中的總人數。然後，每個成員i會選取一個有著m元素的列向量，我們稱它爲內部秘密a_i (internal secret a_i)，接著從這a_i，我們利用$s_i=a_iE \bmod q$的運算可以得出一個稱爲外部秘密(external secret)的s_i值，且此s_i值將會是一個有著n元素的列向量。接著，成員i將這s_i中的第j行的元素，在此稱爲s_{ij}，傳送給成員j。因此，透過列向量a_i與矩陣E的相乘，將只會有少於n的成員數量會從成員i手中得到分享值。

爲了檢驗成員i所分送出去的分享值是確實的秘密分享，此成員必須將含有內

部秘密的承諾值(commitment)公佈出去，來讓大家檢驗是否爲合法的秘密分享；在此，我們就利用由Pedersen於[Ped91a]所提出的可驗證式秘密分享架構，也就是在本篇論文中於 2.3 小節所介紹的可達到資訊安全之VSS架構。因此，成員i便造出另一個內容爲隨機的僞內部秘密a_i'，配合式子$C_i = g^{a_i} h^{a_i'} \mod p$將此承諾向量值廣播出去給所有的檢驗群體$Q_i$，也就是那些拿到非零分享值的成員；接下來，每個在檢驗群體Q_i中的成員都可以去驗證成員i所分享出去的是否是確實含有內部秘密a_i的分享值，而此檢驗方式就是我們於 5.1 小節所述的演算法中之Eq.1。

上述的Eq.1 之檢驗唯有在式子 $s_{ij} = \sum_{k=1}^{m} a_{ik} E_{kj}$ 成立時才能通過。如果檢驗失敗，則表示此分享值是不合法的，此時，成員j將廣播一個針對成員i的“質疑異議”；而爲了回應成員j的質疑，成員i此時必須將s_{ij}及s_{ij}'公佈出去，讓其他檢驗群體中的成員來進一步地檢查是否能通過式子Eq.1，假若依舊不能通過檢驗，此時的成員i將被標示爲“不合法的成員”，因爲他(她)所分享出的分享值必不包含內部秘密，也因此，更不可能透過此秘密分享來重建出金鑰。

透過上述的檢驗過程，標示爲合法成員之集合V將被建立出來，而且每個在檢驗群體Q_i中的成員也將知道是否$i \in V$。在確認誰是合法的成員身分之後，我們就可以透過下述的式子進一步地將秘密x及公開金鑰y給重建出來：

假設 $a = \sum_{i \in V} a_i$ ，又若T是一個線性函數，則 $x = \sum_{i \in V} T(a_i) = T(a)$ ，相對地，公開金鑰也就爲$y=g^x \mod p$

5.2.2 建出公開金鑰

在重建公開金鑰之前，我們必須去防範攻擊者不能藉由操縱重建時的成員去影響或偏頗金鑰重建的分佈，也就是在Gennaro等人於[GJKR99]篇中所提的攻擊情形，但是當合法成員集合V一旦被建立之後，此時的公開金鑰重建過程才能安全地進行。每個屬於V的成員i將他(她)們的相關公開金鑰之分享值廣播出來，也

就是將$A_i = g^{a_i} \bmod p$給廣播出來，此時，其他成員將去檢查其值是否能通過於 5.1 小節演算法中Eq.2 之檢驗。

正如之前的檢驗流程，若檢成員j發現檢驗錯誤，就針對成員i廣播出一個“質疑異議”且成員j爲了證明他(她)的檢驗過程是無誤的，也必須同時將有瑕疵的s_{ij}及s_{ij}’公佈出來；此時，其他成員將重頭地將它們拿來對Eq.1 做檢驗，若是通過，則表示它們確實是從成員i手中分送出去的，接下將再繼續透過Eq.2 之檢驗式子，若成員j所發出的“質疑異議”屬實，則這些成員們將透過廣播他們自己拿到的分享值來重建a_i。而a_i的重建方式就是去解一組與成員i之內部秘密相關的線性等式。

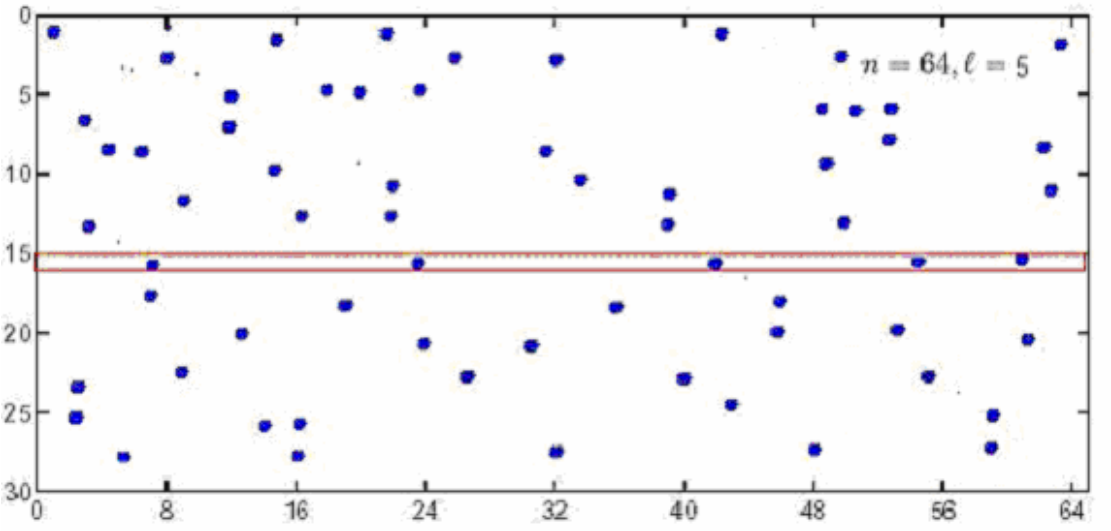
5.2.3 私密金鑰之使用

根據我們對x的定義，當我們知道足夠的x_i時，我們可以透過以下的關係式：$aE=(x_1, x_2, \ldots, x_n)$很直接地找出a。因爲，由 5.1 節以及 5.2.1 小節可知，a是一種與x_i相關的線性函數，也就是我們可以將a寫成$a=(x_1, x_2, \ldots, x_n)\Sigma$，在此的Σ是E的pseudo-inverse。所以，當我們假設$T(a)=av$時，也就有以下的關係：$T(a)=(x_1, x_2, \ldots, x_n)\Sigma v$。因此，假如簽章或是加密爲$g^x$之形式時，每個成員便可以用$g^{x_i}$的形式來簽名，而且這些簽章或加密內容也就可以合併爲$\prod g^{x_i L_i}$之形式，在此的L_i表示Σv中的第i列元素。

5.3 隨機稀疏矩陣

而爲了減少每個成員於分散式金鑰產生之過程所需的通訊量，我們將提出一個隨機稀疏矩陣來取代上一小節中所提之矩陣的角色。而“隨機稀疏矩陣”顧名思義就是一個矩陣中的每個元素之分佈位置是隨機出現的，且此矩陣是個稀疏矩陣；爲了更容易瞭解此矩陣構造，我們將藉著以下頁圖例介紹我們的隨機稀疏矩陣之結構，由圖可知，這是一個行數 n 爲 64，列數 m 爲 29 的矩陣，每一列上將只會有l個非零的值(在此我們以l=5 爲例)，但是這些非零值並不限制必須是連續

出現，可在此列上隨機分佈；因此我們在一開始的矩陣選定上只需對於行數、列數以及非零值的數目多寡做設定，此外並沒有如[CS04]篇中尚須有位移量的額外設定；且值得一提的是在我們的隨機稀疏矩陣中，列的數值選定除了限制不大於行數以外，其值跟行數及非零值的數量多寡完全無關，互相獨立，因此在列值的選擇上有很大的彈性，此值亦攸關門檻是秘密系統中的門檻值，所以我們架構的門檻值可依需求而能做更彈性的調整。



圖六 隨機稀疏矩陣

介紹完上述的隨機稀疏矩陣之後，我們便可以把原先在[CS04]篇中所運用的對角帶狀稀疏矩陣置換爲我們所提出的隨機稀疏矩陣之架構。當把此隨機稀疏矩陣套用於我們在 5.1 小節所闡述的演算法之後，相信在實際運用於網路上大規模成員系統的分散式金鑰產生協定之運作上就能更趨完備！

5.4 主要架構之特性分析

當我們將 5.1 小節的演算法與 5.3 小節的隨機稀疏矩陣相互結合之後，就得出一個$(\gamma-\varepsilon,\gamma+\varepsilon,\delta)$的機率性門檻式之分散式金鑰產生協定架構，其$0<\gamma<\frac{1}{2}$，$\varepsilon>0$，$\delta>0$。

接下來，我們就來探討本架構在實際運用上的實質分析：首先，在矩陣中的行數(也就是 n 值)就代表了所有參與分散式金鑰產生的成員人數，每個成員會依

照自己的索引值(在系統一開始，每個成員便得知自己的所屬序號索引)而隨機地分散於矩陣中的各行數上；矩陣的列數 m，具有與 Pedersen 或 Gennaro 的分散式金鑰產生架構中多項式之次方數相同涵義，也就是說，原先在演算法中，必須有達到某門檻值才能透過 Lagrange 內插法來重建金鑰之原理，但由於一組的多項式能利用凡德蒙地矩陣(詳見於本篇之 3.6 小節)做相對的轉換，因此，在我們將原來的凡德蒙地矩陣置換爲我們的隨機稀疏矩陣後，針對是否能重建出金鑰的門檻亦是以檢查矩陣的秩(rank)是否達到 m(也就是 full rank)，如此一來，就能透過我們於 4.1 小節針對傳統多項式架構的分散式金鑰產生協定改爲矩陣方式之可行性探討中所述的方式來重建金鑰；且值得一提的是，在[CS04]所提之演算法中，每個成員在扮演處理者時，所需隨機選的兩個列向量內容中，雖然只需有 u 個**[註]**元素是非零即可，但此 u 個非零元素卻必須是連續出現，而在我們的架構中，此兩個列向量內容中依舊只需 u 個非零元素，但在此，我們不限定這些 u 個非零元素必須是連續出現，只需隨機分佈即可，因此，在抵擋網路上的叢集錯誤有絕佳的表現！所以，當此列向量與我們的隨機稀疏矩陣相乘運算之後，只會有 $u \cdot l$ 個非零值產生，也就是說當每個成員以處理者身分分送秘密分享時，其所需分送對象只需有 $u \cdot l$ 這麼多即可，如此一來，與以往不論是 Pedersen 或 Gennaro 架構中所需分送訊息必須是所有成員 n 的通訊量相比，在分送的通訊量上更是大大減低其負擔！

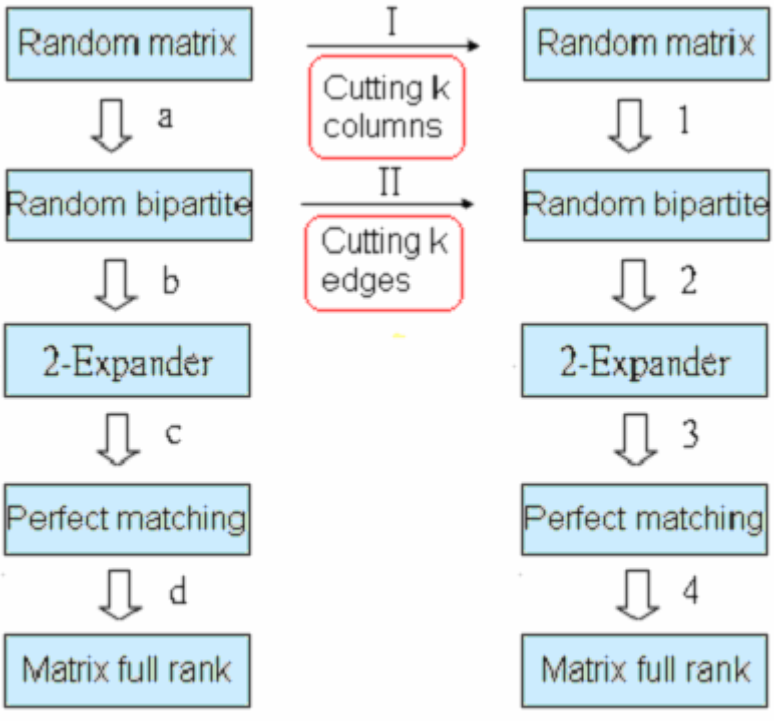
[註] 由於本篇之分散式金鑰產生協定是利用拜占庭協定(Byzantine agreement)之概念來達到廣播之功效，因此，爲了確保所有在檢驗群體(checking group)中的誠實成員能達成多數決的運作方式，我們將把每個列向量中的非零個數 u 設定爲大約是 $\frac{l}{2\varepsilon^2}$ 的數量，而如此設定的依據可以進一步地利用 Hoeffding bound 的機率推導方法(詳見[CS04])，來證明在我們所產生出來的檢驗群體中將有大多數的成員會是誠實的！

5.5 成功率

現在我們探討一個在現實情境中很有可能發生的情形：假設在我們所提出總數有 n 個成員的架構中，含有 αn 個不誠實的成員參與我們建造金鑰的過程(在此我們以 $\alpha = \frac{1}{2} - \varepsilon$ 爲例)，然而誠實的成員有 βn 個(在此的 $\beta n = \frac{1}{2} + \varepsilon$)，接下來我們就來探討在此假設情境中，我們所提出的隨機稀疏矩陣架構能重建私密金鑰的成功機率到底高不高；由我們在之前的敘述說明可知，我們的矩陣架構必須在矩陣 E 爲全秩(full rank)的情況之下，才能重建私密金鑰，因此以下的成功率之證明就是要來探討針對我們的稀疏矩陣 E，在隨機刪除 k 行之後，是否依舊維持全秩，以檢視我們所提出的架構在現實環境中能否實行之可行性。

證明：

(1)爲了證明我們架構的成功率，我們將運用到 random bipartite graph、expander graph、perfect matching 及其與矩陣的全秩與否之間之關聯性，以下就是我們將要證明的檢要流程概述：



圖七 成功機率證明流程

(2)首先，我們先將我們所提出的隨機稀疏矩陣架構對應到我們在 3.1 小節所提的 random bipartite graph，我們將可發現以下的相互關連：

(a)假設矩陣中的每個非零元素以E_{ij}來表示，則列的索引i值就可以對應到 random bipartite graph中左邊端點集合V_1中的端點編號。

(b)相同地，上述E_{ij}中，行的索引j值就可以視爲random bipartite graph中右邊端點集合V_2中的端點編號。

(c)矩陣裡的列數m可應對到random bipartite graph中的$|V_1|$，在此的V_1也就是 bipartite graph中的左邊端點之集合。

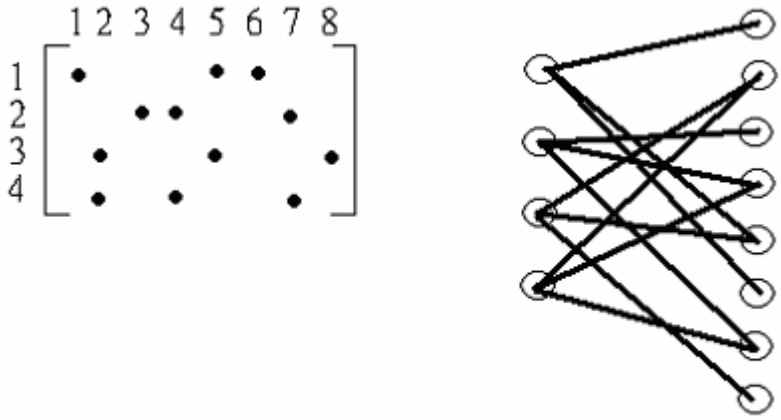
(d)矩陣裡的行數n可應對到random bipartite graph中的$|V_2|$，在此的V2 也就是 bipartite graph中的右邊端點之集合。

(e)每列中d個非零值可應對到random bipartite graph中，左邊V_1集合裡端點的 degree數。

(f)隨機刪去 k 個行數，也就是排除 k 個不誠實成員後，可視爲對 random bipartite graph 隨機刪去 k 條相連的 edge 數。

(g)因此，我們有了成功率流程圖中的“a”及“1”之關聯結果。

(h)下圖是以一個尺寸爲 4 x 8 且每列有三個非零值的隨機稀疏矩陣與 random bipartite graph 之間的對應關係之例子：



圖八 隨機稀疏矩陣與random bipartite之對應

(3)有了上述隨機稀疏矩陣與 random bipartite graph 之間的對應關係之後，我們可以推知以下關係：

(a)當我們對一個隨機稀疏矩陣“隨機地”刪去 k 行之後，還是呈現一個隨機稀疏矩陣。(也就是成功率流程圖中的“I”)

(b)當我們對一個 random bipartite graph“隨機地”刪去 k 條 edge 之後，還是呈現一個 random bipartite graph。(也就是成功率流程圖中的“II”)

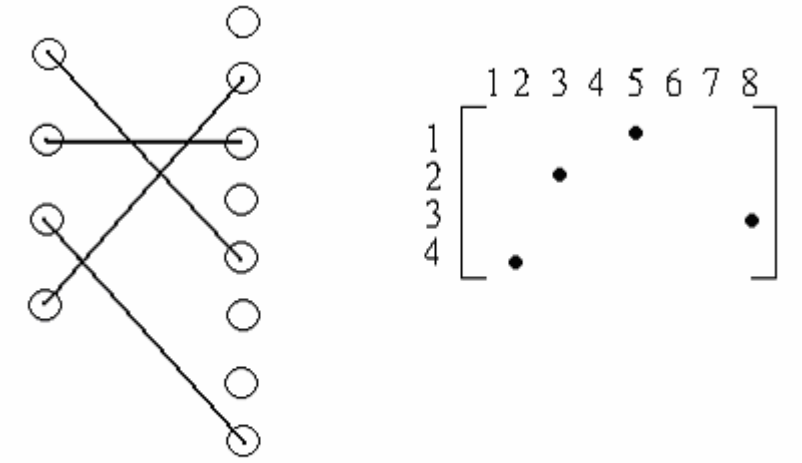
(4)由 3.3 小節的敘述可知，我們的隨機稀疏矩陣經過應對後的 random bipartite graph 有幾乎趨近於“1”的機率(約爲$1-2[4e^3(\frac{4}{n})^{d-3}]^2$)可以成爲一個 2-expander bipartite graph；因此，證明流程圖中的“b”及“2”步驟即可推知！

(5)當我們得出我們的矩陣架構有非常高的機率成爲 2-expander bipartite graph 之後，我們接下來便可以運用在本篇 3.5 小節所介紹的一個 bipartite graph 與 perfect matching 之間關聯性的 Hall Theorem，推導出我們所形成的 2-expander bipartite graph 也會形成一個由左端點集合到右端點集合的 perfect matching。從 Hall Theorem 中可知，一個 bipartite graph 若要成爲一個 perfect matching 之條件必須是$|\Gamma(S)| \geq |S|$，也就是從左方任意端點之集合所連出的右方鄰居

端點集合之尺寸要大於等於左方端點之集合。而從本小節的第(4)點說明可知，我們所形成的 2-expander bipartite graph 有“$|\Gamma(S)| \geq 2|S|$”的關係；因此，我們可以得出我們架構形成的 bipartite graph 將會有 perfect matching，所以證明流程圖中的“c”及“3”步驟亦可推知。

(6)當我們得出一個 2-expander bipartite graph 將是一個 perfect matching 圖形之結果後，我們就利用以下的方式來還原爲矩陣型態，並進一步討論此被還原的矩陣是否依舊維持全秩(full rank)：

(a)我們將此 perfect matching 的圖形之左端點集合中的端點相對應到矩陣的列索引數，而圖形中的右端點集合中的端點對應到矩陣的行索引數，因此，我們便可以將矩陣中的非零點標示出來。可由下頁圖例得知：



圖九 perfect matching還原爲矩陣

(b)又因爲由 perfect matching 的性質可知，每個左端點接會與一個跟其他左端點不重複的右端點相連，這就隱含著：在矩陣中的所有行中，將有 m 個行與 m 列數一對一相連，也就是此行的最大獨立數將等於 m，也等於列的最大獨立數，因爲列數爲 m，所以最後還原出的矩陣依舊是全秩(full rank)

(c)值得注意的是，上述的(b)情況必須在“每個非零值之選取皆不重複”之額外條件之下才能成立，否則，只要將此矩陣透過高斯消去法的調整之後，必

然會有秩(rank)減少的情形發生；而從 5.1 小節的演算法可得知，因爲我們的稀疏矩陣E中的非零值之元素是從Z_q之中所選出的，且q是一個很大的質數值，所以兩個非零值會重複的機率非常的低。

(7)因此，我們推倒出成功率流程圖中的“d”與“4”之結果。

(8)透過上述一連串的推導過程，我們將得出：當我們將我們的隨機稀疏矩陣隨機刪去 k 行之後，仍然有非常高的機率會維持全秩，也因此，能有非常高的機率重建出私密金鑰！

5.6 正確性

(1) 針對(C1)：

由於透過廣播的資訊，在檢驗群體(checking group) Q_i中的誠實成員可以來判斷並決定成員i是否是在合法成員V集合中；至於其他在檢驗群體之外的誠實成員則信賴在檢驗群體中大多數人對於成員i的決定結果。然後，經由演算法中步驟三所產生出的合法集合成員是唯一的，假如在每個檢驗群體中絕大部分是誠實的成員，則此機率是非常高的。

在假設攻擊者不可能算離散對數的情況下，在演算法中的檢驗式子Eq.1 隱含著成員的分享值對於同個a_i是一致的。而由 4.1、5.1 及 5.4 小節的相關說明及分析可知“重建原理”隱含著在任何集合中誠實成員的列會是全秩(full rank)，所以他們有很高的機率是符合唯一的a_i

因此，所有通過檢驗式子Eq.1 之帶有分享值的誠實成員可以重建私密金鑰。

(2) 針對(C2)：

當有任何針對成員i的合法“質疑異議”(complaint)被提出時，相對的這個a_i會被公開地再透過重建原理來重建它，而且$A_i = g^{a_i}$。

現在，我們再考慮當沒有任何的質疑異議被提出時，因爲a_i是可以由那些握有分享值的誠實成員重建而造出，因此，至少會有u個線性獨立的等式以及u

個未知數，所以將得出一個唯一的解。

從成員i廣播出的這些等式，透過演算法中的檢驗式子Eq.2 的檢驗，可以清楚得知所廣播的A_i就是 g^{a_i} 。因此，所有在每個檢驗群體中的誠實成員會得有正確的A_i，$i \in V$，而且，因爲在每個檢驗群體中有多數的誠實成員，因此所有誠實的成員會拿有相同的A_i，$i \in V$且$y=g^x \bmod p$

(3) 針對(C3)：

我們定義的秘密值是$x=(\sum_{i \in V} a_i)v$ ，因爲v是隨機選取的，且任何隨機值a_i是跟a_j獨立的，如此一來，會導致x是個隨機分佈的值。

(4) 針對(C4)：

我們可以從“重建原理”及 5.5 小節中所探討的成功機率中，很直接地得出此結果。

5.7 安全性

本小節針對我們所提出的架構所探討的相關安全性證明方式是：我們將造出一個能成功模擬出任何一個誠實的使用者與實際握有秘密的處理者，在透過之前所述的正常合法演算法時，他們兩者之間的所有流程及相互通訊之訊息的模擬器(simulator)，藉此來證明惡意攻擊者(adversary)不能得到任何跟私密金鑰 x 相關的資訊，除非此攻擊者能解公開金鑰 y 的離散對數。而且，我們所造的模擬器是屬於一種機率性多項式時間(probabilistic polynomial-time)的演算法。

我們將要模擬的大致手法如下：我們給定$y \in Z_p$，也就是$y=g^x \bmod p$，模擬器可以產生出一個跟經過正常執行協定之後，所輸出的此y值之機率分佈狀況之差別是無法分辨的。

也因爲我們所運用的演算法是利用隨機的方式去分佈攻擊者，因此我們的模擬器會有很高的機率模擬(simulate)成功。另外，在我們的模擬過程中，是假設惡

意攻擊者不能控制超過($\frac{1}{2}-\varepsilon$)的成員。而我們的模擬器之輸入是可以透過正常協定所產生出來的 y 值。

假設被惡意攻擊者控制的成員集合爲$B=\{B_1, B_2,\ldots, B_t\}$，而誠實的成員爲$G=\{G_1, G_2,\ldots,G_{n-t}\}$。

接下來我們來看看以下模擬器的演算法，SIM：

(1) 每個誠實成員 i∈G，執行正常演算法中的步驟 2 及步驟 3，因此：

(a) 集合 V 就被定義出來，並且 G⊆V。

(b) B可以看到的資訊有a_i及a_i’ 對於i∈B，s_{ij}及s_{ij}’ 對於i∈V且j∈B和C_{ij}對於i∈V。

(c) SIM知道a_i及a_i’ 對於所有i∈V。

(2) 然後做以下的運算：

(a) 算出$A_i= g^{a_i} \bmod p$對於i∈V \{G_i}。

(b) 設S是Q_{G_1}的一個子集合，也就是G_i的檢驗群體，它包含了所有$Q_{G_1} \cap B$以及足夠的$Q_{G_1} \cap G$，以至於El_S的秩是u-1。

(c) 設 r 是某個 i，而讓 S∪{r}的秩爲 u。

(d) 給定$s^*_{G_{1j}} = s_{G_{1j}}$對於 j∈S。

(e) 設 S’ 是含有 S 中 u-1 個元素的子集合，且秩爲 u-1。

(f) 計算Σ，也就是 E 的子矩陣S'∪{r}之反矩陣(inverse)。

(g) $a^*_{G_1} = (s^*_{G_1S_1'},\ldots, s^*_{G_1S'_{u-1}}, s^*_{G_1r})\Sigma$，但是$s^*_{G_1r}$的值還尚未固定；

同樣地，我們假設$A^*_{G_1} = (g^{\alpha_1+\beta_{1S^*_{G_1r}}},\ldots, g^{\alpha_u+\beta_{uS^*_{G_1r}}})$，而其中的$\alpha_i$及$\beta_i$是Σ相關的函式構造(function)而且$s^*_{G_1j}$ for j∈S’ 。

(h) 因爲$\prod_j A^{v_j}_{G_{ij}} = y\left(\prod_{i\in V\setminus\{G_1\}}\prod_j A^{v_j}_{ij}\right)^{-1}$，而等式右邊展開之後可以化簡爲一種 g 的某次方形式，又等式左邊可化簡爲 g 的一種$\alpha+\beta s^*_{G_1r}$之形式，接著我們對

等式兩邊的次方進行簡單的移項動作就可以找到 $g^{s^*_{G_1 r}}$ ，然後進一步地也可得出 $A^*_{G_1}$ 。

(i) 接著，我們將A_i for i∈G\$G_1$和 $A^*_{G_1}$ 廣播出去。

(j) 對於每個i∈G的成員的A_i for j∈B執行原演算法中步驟 4.(b)的檢查式子，然後廣播出任何必要的“質疑異議”。

(k) 因爲我們是依照正常的演算法步驟執行，所以惡意攻擊者不可能對任何誠實的成員提出任何合法的“質疑異議”；而且模擬器必須進一步地復原所有惡意攻擊者的干預動作，也就是說，模擬器將可利用演算法中的步驟 4.(c)來復原a_i(就是重建任何成員沒有適當地分享出A_i)。

(l) 因爲對於i∈G\$G_1$，$a_i$及$a_i$’ 是依據隨機均勻分佈而選的；相同的， $a^*_{G_1}$ 的形成也是隨機的；而透過 $C_{G_1} = g^{a^*_{G_1}} h^{a'^*_{G_1}}$ 這公開的資訊，可由 $a^*_{G_1}$ 及 a'^*_i 之間的關係得出： $a'^*_{G_1} = d \log_g(h)^{-1} \cdot (a_{G_1} - a^*_{G_1}) + a'_{G_1}$ ，這也可說明 $a'^*_{G_1}$ 也是隨機的。

5.8 與之前所提出架構之比較及貢獻

除了保有 J. Canny 等人於 [CS04]篇中所提架構之優點：所需分享出去的數量比傳統的分散式金鑰產生協定少很多、可實際運用於大規模成員架構上、符合 Gennaro 等人在[GJKR99]所訂出的較嚴謹的安全定義；除此之外，我們針對[CS04]篇中尚有的不足處，提出以下的改進及貢獻：

(1) 當 J. Canny 等人的 Practical Large-scale DKG 中所用的對角帶狀稀疏矩陣之每個元素即使是隨機排列時(也就是其矩陣 E 的行數所對應的成員是隨機的，成員之間沒任何關聯，其矩陣中非零值的帶狀結構雖是連續出現，亦不受網路的叢集錯誤影響或不誠實成員之連續出現的干擾影響時)：

(a) 因爲每個人都知道E(也就是對角帶狀稀疏矩陣的構造是公開的)，而且每個a_i中連續u個非零值之起始位置規則也是大家都知道的(詳見[CS04])；因此，

若透過一群惡意成員的合作，便可推知所剩下的成員中哪些是誠實的，進而算出a_i的非零起始位置，再配合所知的E，兩者結合之後，就可以推知這些誠實成員的驗證群是哪些，然後惡意攻擊者就可以進一步地對那些要接收分享值的接收者作一些干擾或破壞的動作，如：阻斷服務(DoS)或假扮等等。

而本篇所使用的隨機稀疏矩陣，因爲矩陣中的每個元素都是隨機出現的，所以能避免此情況發生。

(b) 另外，在演算法中有兩個需透過廣播之值($C_i = g^{a_i} h^{a_i'}$ 及$A_i = g^{a_i}$)是跟a_i有關，而因爲在a_i的結構中有u個“連續”非零值，因此在傳送此兩值的過程中，內含連續數值的a_i之完整性很容易受到網路上的叢集錯誤之影響，進而影響到此兩個透過網路方式的廣播值之正確性。

而本篇所使用的a_i內容值是不須連續出現，是隨機分佈的，因此較能避免此情況發生。

(2) 在 J. Canny 等人於[CS04]所提的 Practical Large-scale DKG 中所用的對角帶狀稀疏矩陣之各個元素若非隨機安排時，也就是其矩陣 E 的行數所對應的成員間出現的順序關聯性，除了會發生上述(1)中所提到的兩個主要弱點之外；在此非隨機安排的影響下，亦將導致其矩陣中連續出現非零值的帶狀結構相互之間有相對應連續出現的順序上關聯，同樣地，在網路上易發生的叢集錯誤(burst error)也極容易發生在此對角帶狀稀疏矩陣的傳送上，進而影響它的完整性；且若不誠實的成員恰巧連續出現 l 個，若受其干擾或將此 l 個不誠實成員全數屏除時，將會導致矩陣 E 的秩數減低，若非全秩，此重建金鑰的過程也終將失敗！

而因爲本篇所使用的是隨機稀疏矩陣，每個元素都是隨機分佈的，因此較能避免此情況發生。

(3) 在 J. Canny 等人於[CS04]所提的 Practical Large-scale DKG 中，因爲帶狀稀疏矩陣之列的大小(也就是 m 值)會因爲行(n 值)之大小、矩陣中帶狀之長度及位移

多寡之影響，讓 m 值因此三值而固定住；然而，我們隨機稀疏矩陣中的列數之大小不需固定！如此的特性將帶來什麼優點呢？因爲矩陣中的列數直接牽涉到在我們透過 Lagrange 內插法解 t 次方程式，來重建私密金鑰所需的密碼系統門檻值 t。

由此可知，我們的密碼系統門檻值可因我們所需的安全程度需要，做更有彈性的調整。

第六章 實驗結果之比較及分析

為了驗證我們所提出的架構之實用性，我們根據[CS04]篇中所述的架構以及本篇所提的架構，利用撰寫程式的方式，分別實作出兩套系統，再將相同的輸入數據所實驗出的不同輸出結果，進一步地來做更實際的差異度比較。

6.1 實驗數據及結果

首先，我們先利用較小的行數為輸入，做多方面參數設定的調整，來觀察[CS04]中所提出的架構之一些特性：

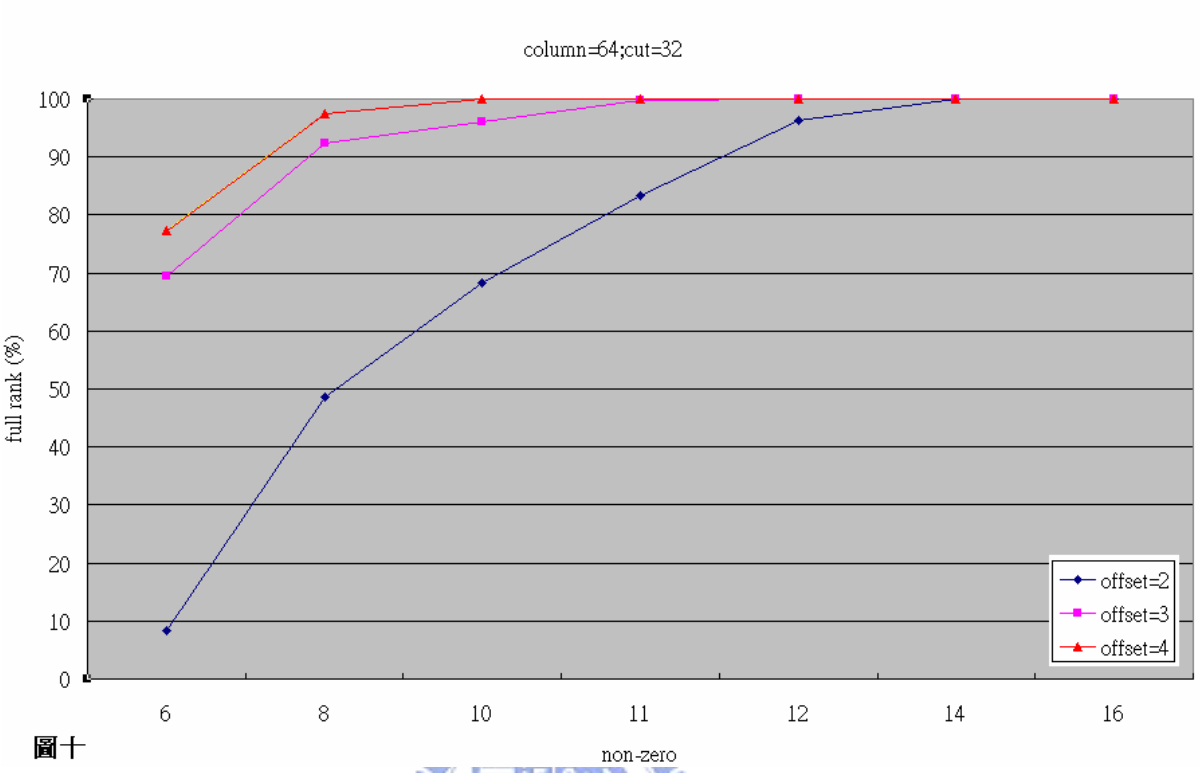
由 $m = \frac{(n-l)}{offset}$ 可得知，當我們將原來的總行數及所刪去的行數固定時，我們可以發現在對角帶狀稀疏矩陣中，列與列之間非零值的位移量(offset)以及每列非零值的個數皆會影響此矩陣在任意刪除行數之後能否達到全秩(full rank)，也就是能否重建金鑰的關鍵因素，因此接下來我們將分別探討位移量及每列非零個數的影響：

由下頁圖十可知，當我們除了固定上述的總行數及刪去行數之外，我們亦將每列的非零值個數固定為某值之後，我們可以發現：位移量每增加一點點，矩陣能達到全秩的機率將會很快速地向上提升，這是因為[CS04]中的位移量大大影響了矩陣中的列數(m)，也就是有 $m = \frac{(n-l)}{offset}$ 的關係。

下頁的圖十一所示是假設總行數、刪去行數及位移量固定的前提之下所做的實驗，在此，為了能明顯看出每列非零值個數的影響，我們將位移量設為較小的數“2”，我們將可以發現隨著每列非零值個數的增加，矩陣達到全秩的機率也相對提升，但是所提升的速率與位移量變化比較之下較為緩慢。

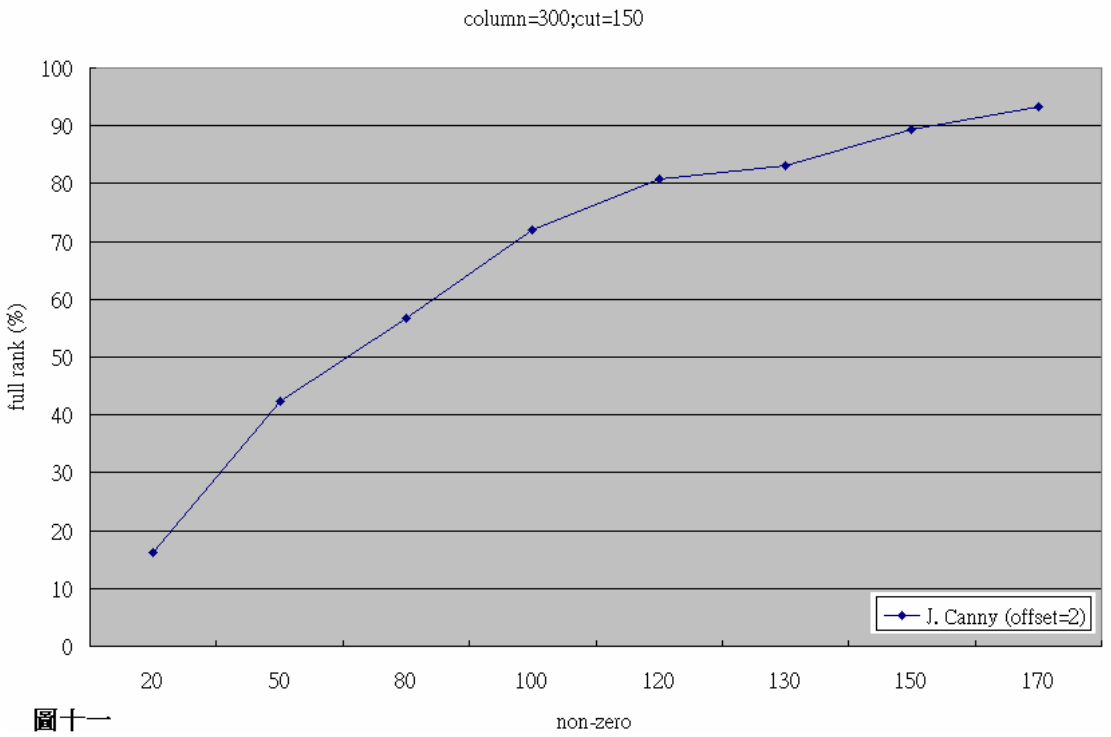
總結上述兩論點，可發現；當位移量增大或是每列的非零值之個數增加時，皆能影響相對的列數(m)，而使之相對變小，由於行的最大獨立數等於列的最大獨立數之秩的性質，因此在所需的全秩也將相對變小，在固定刪去行數的情況

下，達成全秩的機率也就相對增大許多！



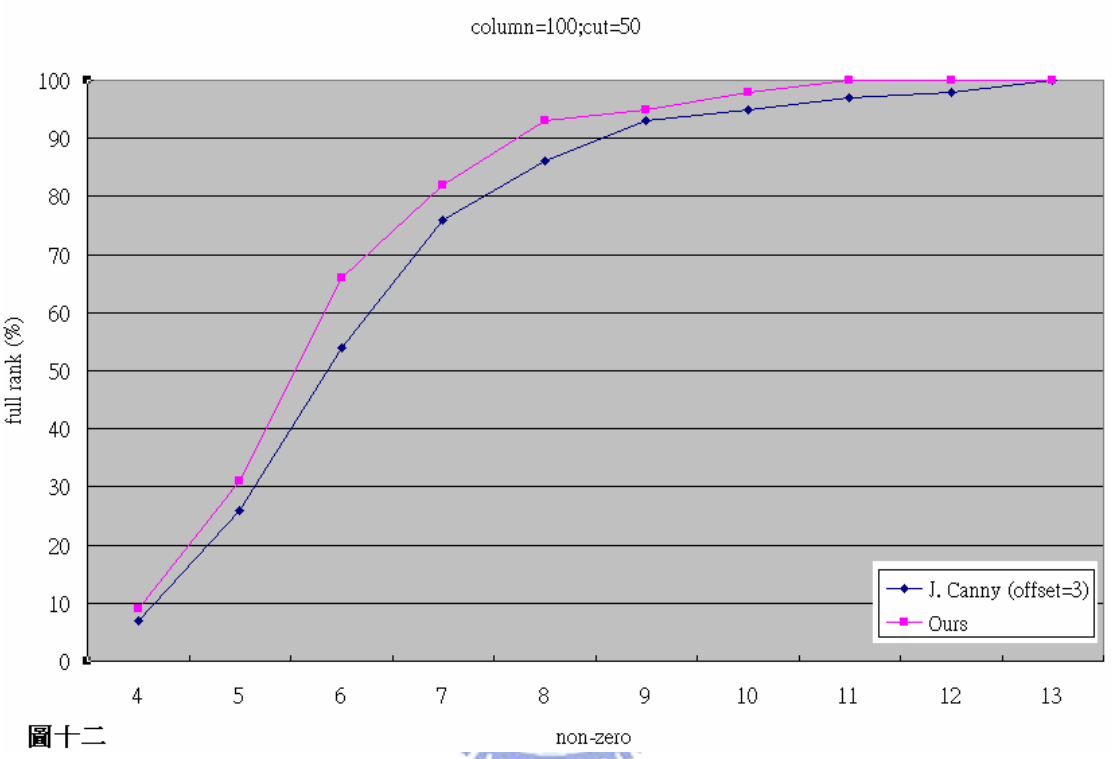
圖十

由圖十的實驗結果觀察得知，在位移量爲 3 以上時，不需要數量很多的每零值，便能使此任意刪除行數之矩陣達到全秩的機率很快地超過 90%，但因爲位移量(γ^{-1})是位於分母，可想而知，除法的影響遠超過l在分子中的減法。



圖十一

由上頁的圖十實驗結果得知：當[CS04]的位移量在 3 以上時，不需很多的每列非零值個數即可達到不錯的全秩機率表現；在此，我們將以[CS04]架構在位移量爲 3 時的情況下，來與我們所提出的架構於相同環境參數下，透過實驗過程中作一個較初步的比較；如下圖所示：

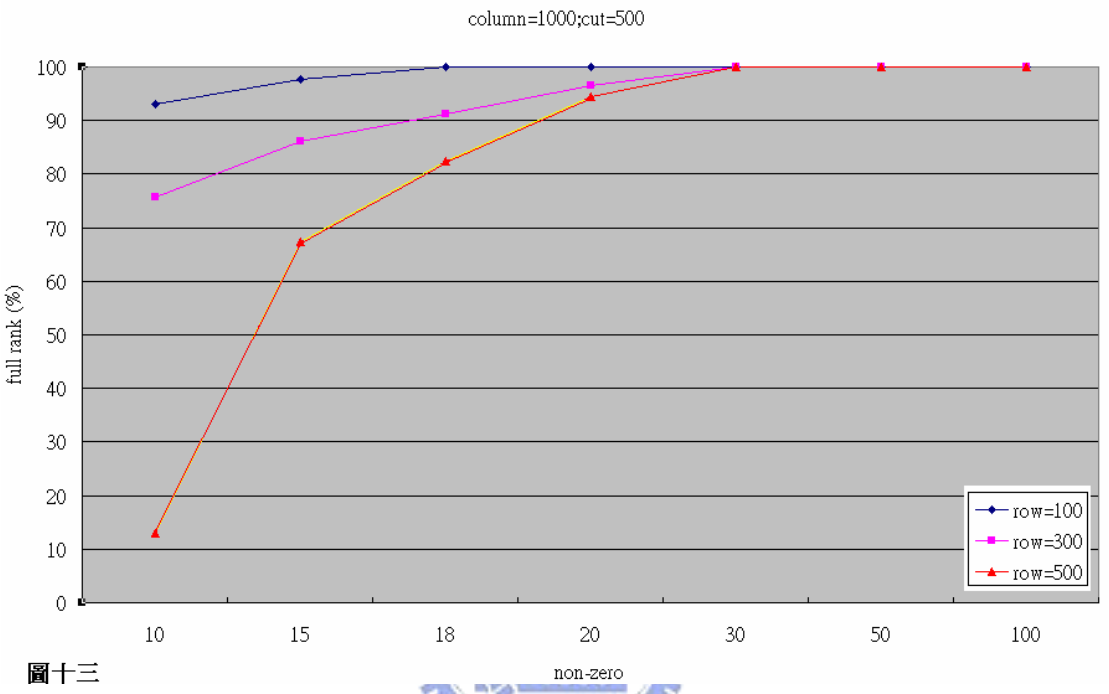


圖十二

因此，我們從實驗結果將可以發現，在成員人數很少時(也就是 n 值取的比較小時)，我們的架構(圖十二中標示爲 Ours 的曲線)會比[CS04]的帶狀且位移量爲 3 曲線之達到全秩的機率表現來的更好！那我們不禁會去思索，當成員人數很大，也就是 n 爲上千或上萬時，我們的矩陣架構爲了達到高機率的全秩，是否每列所需的非零值依舊會小於[CS04]之所需呢？

有了上述兩種架構的一些主要特性基本分析的概念之後，現在我們就可以進入我們以下的主要實驗主題：由於我們本篇探討的是在大規模成員參與之下的分散式金鑰產生協定，所以，在接下來的實驗我們將把我們的成員參與人數提高，然後去檢視並探討我們所提出的架構在大規模成員參與的情況之下，各參數之間相互影響的關係爲何？達到全秩的機率是多少？在什麼樣的參數調整及設定之下才能有很高的機率達到全秩？

下頁圖十三所將呈現的是當我們所提出的隨機稀疏矩陣架構在成員爲 1000 人時(也就是行數 n=1000)，且受到惡意攻擊者所控制的人數爲 500 人時(也就是必須任意刪除 500 行)，在不同的列數變化(也就是在不同的金鑰重建門檻)之下，所相對應的每列非零值個數關係圖。



圖十三

由圖可知，在行數爲 1000 且必須任意刪去的行數爲 500 時(總人數的一半)，不論是列數較小的 100 或剛好爲總行數之一半的 500 列(在此我們設定列數最大不超過總行數的一半)，要使矩陣達到全秩的機率超過 90%時，每列非零值的個數皆只需比 20 個數量少即可達成；這也就是說，當有 1000 個成員利用我們所提出的隨機稀疏矩陣架構來做分散式金鑰之產生時，即使有半數，也就是 500 人遭受到惡意攻擊者的控制之下，我們的矩陣架構只需極少的非零值就有非常高的機率能達到全秩，並進一步地將金鑰重建回來，且在重建金鑰的安全門檻(也就是矩陣中的列數)可以很有彈性地依安全需求作調整及設定！

6.2 實驗結果之比較

依照[CS04]的作者針對達到全秩機率超過 90%，且有著大規模成員參與之情況之下，所需 l 的理論數值建議如下：當總人數 n=1000 時，l 約需 170。

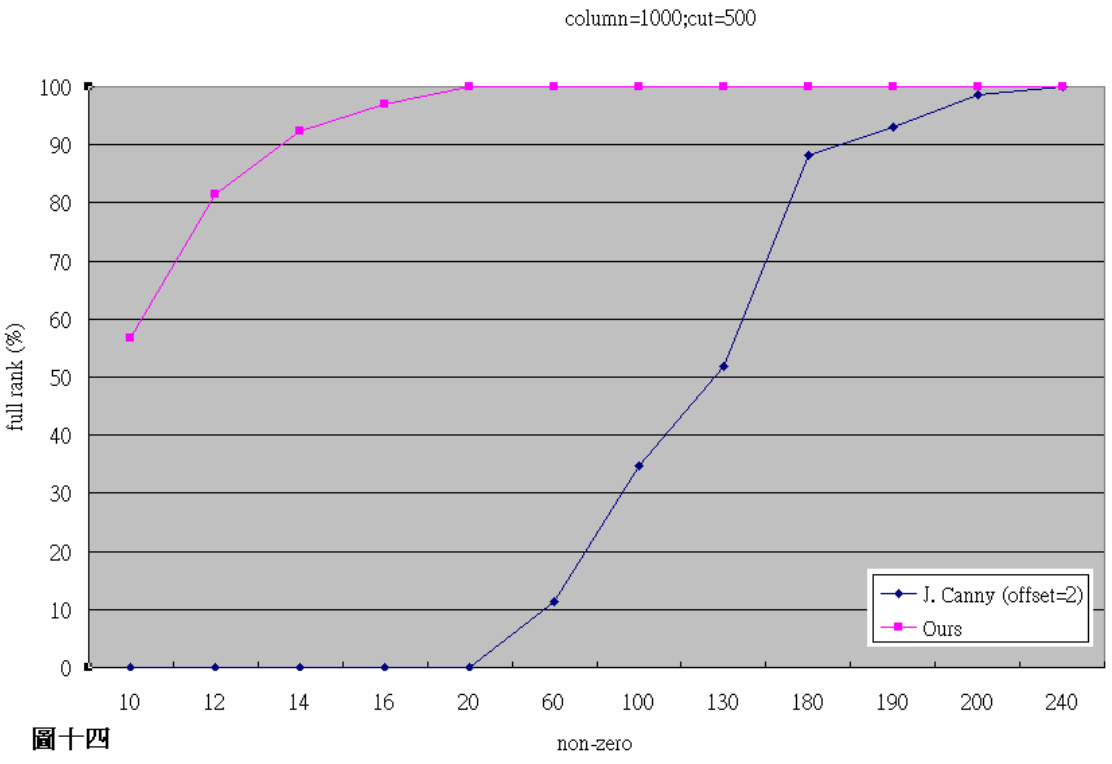
但是，經我們實際運用[CS04]之演算法所架構出的 DKG 系統來進行實驗，卻發現在實驗時所需的 l 值將異於其理論估計值(因爲亦受位移量之大小所影響)，因此我們將針對這些細部的參數做適當的調整來模擬我們的實驗比較數據。

下述的對照表所將呈現的是針對在[CS04]中的兩種不同的 offset 調整值所實驗出來的數據，並將此兩組數據與我們的架構做較深入的比較。而此對照表格之假設前提爲：假設系統總人數 n=1000，我們將探討在隨機刪去總行數的一半之後，矩陣依然會達到全秩的機率有 90%時之比較結果。

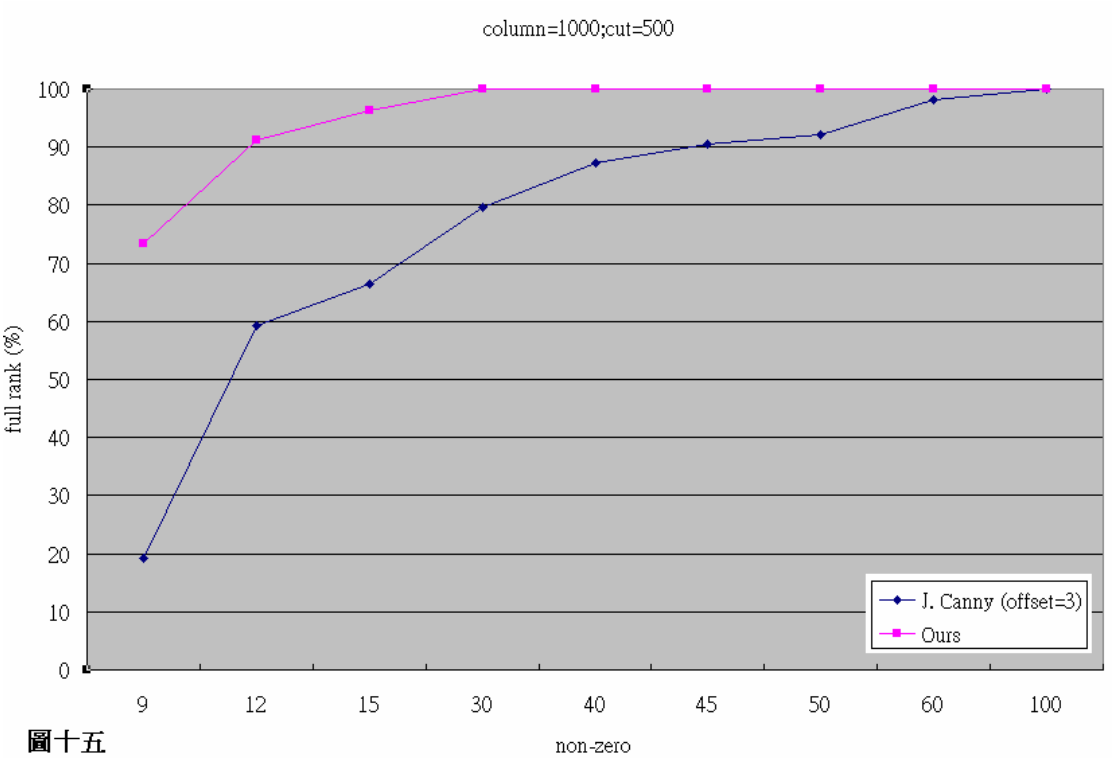
	列數爲 **408** 時 (此時的 offset=**2**)	列數爲 **318** 時 (此時的 offset=**3**)	列數爲 **242** 時 (此時的 offset=**4**)
J.Canny [CS04]	對角帶狀稀疏矩陣中的每列非零值之個數約需 **185** 個	對角帶狀稀疏矩陣中的每列非零值之個數約需 **45** 個	對角帶狀稀疏矩陣中的每列非零值之個數約需 **33** 個
Ours	隨機稀疏矩陣中的每列非零值之個數約需 **14** 個	隨機稀疏矩陣中的每列非零值之個數約需 **11** 個	隨機稀疏矩陣中的每列非零值之個數約需 **8** 個

PS.上述之列數是爲配合[CS04]架構所調整出來的。

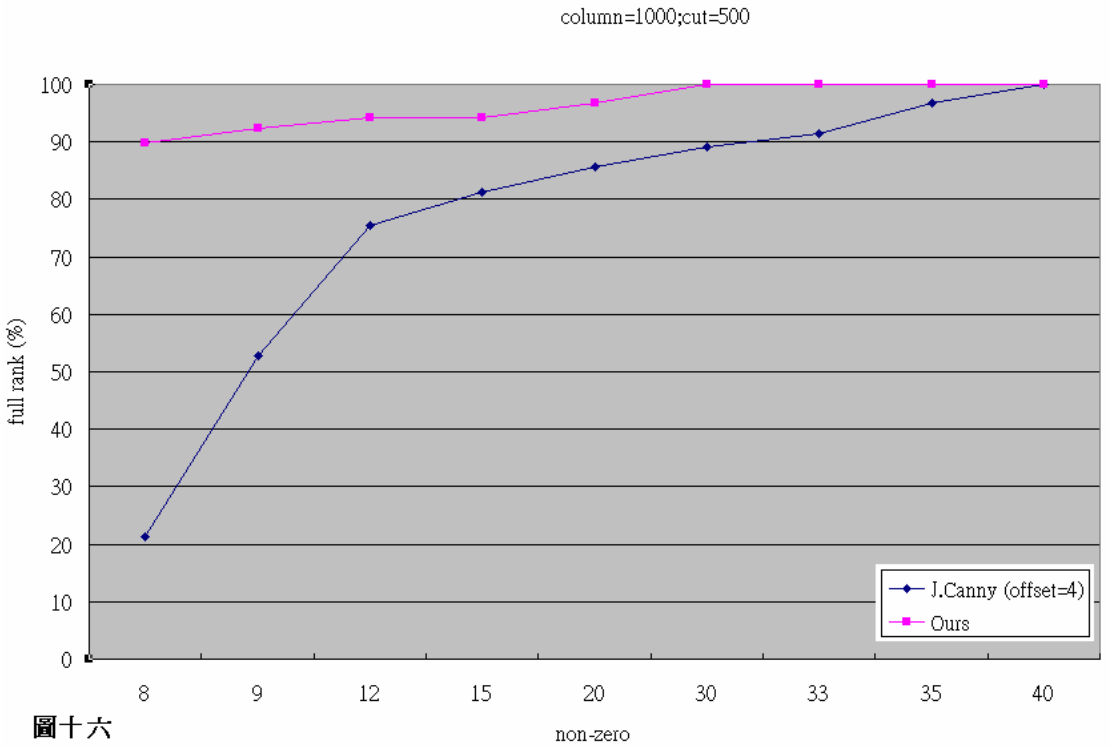
由上表之比較結果可知，在相同的列數假設下，我們的架構在所需的 l 值上遠小於[CS04]篇中之所需。



圖十四



圖十五



圖十六

6.3實驗結果之分析

在[CS04]篇中，作者針對每人在整個秘密分享時，所需的通訊人數之數量上給了一個大約的估計式子：$l+\gamma^{-1}\cdot u$，在此的γ^{-1}爲位移量；而此處的 u 是在 5.1 節所提之演算法中，用來與稀疏矩陣 E 相乘的列向量中所含的非零值之個數；

而上述的估計式子，是由列數、行數、位移量及每列非零值之個數與此作者所提出的對角帶狀隨機稀疏矩陣相互運算，將運算之後非零的值當作所需分享出去的通訊對象而得出的估計值。

而在本篇論文中，由於沒有位移量此變數作用的影響，我們架構中每人所需通訊的對象之數量也就大約演變爲 u • l，也就是在 5.1 節之演算法中，列向量中 u 個非零值與我們的隨機稀疏矩陣相成運算後，最多有 u • l 個非零的值產生，而就把這些非零值標示爲每個人所需秘密分享通訊的對象。

若只單單比較上述兩方的所需通訊數量式子，乍看之下，我們的通訊量與[CS04]篇中所需的通訊量相差甚多；但是，從 6.2 小節中的每列所需非零值個數之比較對照表可知，針對相同數量的大規模成員而言，我們所需的 l 值遠小於[CS04]篇中之所需！且根據實驗結果，[CS04]的對角帶狀稀疏矩陣在成員數量越大時，若要很容易達到高機率的全秩，主要可以調整的參數有兩項：

(1) 其一是“位移量(offset)”，而位移的量(γ^{-1})越大，相對的列數將極劇減小，因此，當較小的列數在我們的架構運作時，我們所需的 l 值將會非常的小；

(2) 另一調整方式爲“每列中的非零值個數”，也就是 l 值，因此若固定列數而要讓矩陣達到全秩機率爲 90%時，[CS04]所需的 l 將會遠大於我們架構所需的；

簡而言之，不論是做上述的任何一種調整，我們所需的 l 值將遠遠小於[CS04]之所需！

另外，在 5.4 小節中，我們知道 u 的值大概是 $\frac{l}{2\varepsilon^2}$，而在此的 ε 是個介於 0 到 1/2 的數(詳見本篇文中的 5.4 小節)由此可知，此處的 u 值將受到 l 值大小之影響，所以我們架構中所需的 u 值也會相對地比[CS04]篇中所需的還要小很多；

最後，從整體來看我們的實際通訊量 u’• l’與[CS04]的 $l+\gamma^{-1}$ • u，由於 u’遠小於 u 且 l’遠小於 l，因此可推知：“我們架構所需的通訊量並不比[CS04]所需

的來的遜色！”；底下，我們將加入實際實驗數據，以更深入的角度來檢視上述的論點之正確性：

當把 u’=$\frac{l'}{2\varepsilon^2}$ 值代入我們的所需通訊量 u’・l’，經化簡可得：$\frac{(l')^2}{2\varepsilon^2}$

當把 u=$\frac{l}{2\varepsilon^2}$ 值代入[CS04]架構所需的通訊量l+u・offset，經化簡可得：

$\frac{l(2\varepsilon^2+offset)}{2\varepsilon^2}$

有了上述我們兩方所需通訊量的化簡式之後，我們就把在 6.2 小節中所提到的實驗比較數據分別代入式子中，發現：

(1) 當[CS04]在總人數爲 1000 且位移量爲 2 時，在能抵擋一半人數的惡意攻擊者還有 90%重建金鑰成功的情況下，他們所需的通訊量約爲$\frac{185(2\varepsilon^2+2)}{2\varepsilon^2}$，而我們的架構所需通訊量約爲$\frac{14^2}{2\varepsilon^2}$，在此的$\varepsilon$是個介於 0~1/2 的數。在本例中我們的所需通訊量少於[CS04]。

(2) 當[CS04]在總人數爲 1000 且位移量爲 3 時，在能抵擋一半人數的惡意攻擊者還有 90%重建金鑰成功的情況下，他們所需的通訊量約爲$\frac{45(2\varepsilon^2+3)}{2\varepsilon^2}$，而我們的架構所需通訊量約爲$\frac{11^2}{2\varepsilon^2}$，在此的$\varepsilon$是個介於 0~1/2 的數。在本例中我們的所需通訊量略少於[CS04]。

(3) 當[CS04]在總人數爲 1000 且位移量爲 4 時，在能抵擋一半人數的惡意攻擊者還有 90%重建金鑰成功的情況下，他們所需的通訊量約爲$\frac{33(2\varepsilon^2+4)}{2\varepsilon^2}$，而我們的架構所需通訊量約爲$\frac{9^2}{2\varepsilon^2}$，在此的$\varepsilon$是個介於 0~1/2 的數。在本例中我們的所需通訊量略少於[CS04]。

(4) 當我們將總人數再調高時，經實驗結果得知：[CS04]在總人數爲 1500、位移量爲 3 且能抵擋一半人數的惡意攻擊者還有 90%重建金鑰成功的情況下，所

需的每列非零值約爲 55，所以代入所需通訊量式子 $\frac{l(2\varepsilon^2 + offset)}{2\varepsilon^2}$ ，得出約爲：$\frac{55(2\varepsilon^2 + 3)}{2\varepsilon^2}$ ，而我們架構所需的通訊量約爲 $\frac{16^2}{2\varepsilon^2}$ 。在此例中，我們所需的略大於[CS04]。

從以上的實際實驗比較可發現，當總人數越趨變大時，我們比[CS04]架構所需的通訊量也將變大，但因爲我們所需的每列非零值遠小[CS04]，所以我們額外多出的所需通訊量不致於被拉大許多。

因此，我們所提之架構中，每個成員在分散式金鑰產生的過程中所需的通訊數量雖然沒有比[CS04]篇中所需通訊量上之表現來的出色，但是，每人所需通訊量在實際的實驗數據上與[CS04]相比之差距並不多。

然而，正如我們在 5.8 節中所論述的，我們的架構在實際運用於資訊網路的現實應用層面上，有著更大的潛力，其主要因素有：我們的架構對於資料間於網路上之傳輸更爲堅韌保護、對於安全性參數之設定更有彈性且在分散式金鑰產生的過程中，提供更安全的保障！

於通訊數量上的效率耗費與資料間安全及完備性上取捨之際，我們所提出的架構在實際運作於網際網路中的大規模成員參與情況之下，不失爲一個出色的分散式金鑰產生之方案！

第七章 結論

在本篇論文中，我們先介紹一系列可驗證式秘密分享協定(VSS)與分散式金鑰產生協定(DKG)之相關技術歷史演進，並進一步地引導出由 J. Canny 在 Eurocrypt 2004 中所發表的[CS04]，它是目前已知，在所有已提出的分散式金鑰產生協定中，且不失其嚴謹安全性之下，最能符合大規模成員參與之系統架構，其突破性的利用稀疏矩陣之特性，將成員間所需分送的通訊量大大降低許多，此構想之前瞻性備受矚目！

但由於資訊網路之便利，使得許多技術應用也須因應網路世界而有所革新，當然，分散式金鑰產生協定也受此衝擊，因此，當我們仔細探究 J. Canny 的以對角帶狀稀疏矩陣之分散式金鑰產生協定時，卻發現其架構在實行於網路之應用上會有許多不妥及待改進之處；藉於此需求，我們提出一個以隨機稀疏矩陣置換其對角帶狀稀疏矩陣之分散式金鑰產生架構：在通訊量之效率上不亞於 J. Canny 所提出的架構，但在抵擋網路上的叢集錯誤或惡意攻擊者的阻斷服務上有更不錯的表現。

此外，我們的架構除了能提供非常高的成功機率以達到分散式金鑰產生之目的，並能證明我們的架構能有不洩漏任何資訊的安全性之外，亦能符合由 Gennaro 等人於[GJKR99]針對分散式金鑰產生協定所提出的嚴謹安全定義。

第八章 參考文獻

[Sha79] Adi Shamir: How to Share a Secret. *Communication of the ACM*, volume 22 number 11, pages 612-613. (1979)

[Fel87] Paul Feldman: A Practical Scheme for Non-interactive Verifiable Secret Sharing. In *Proceedings of 28th Annual Symposium on Foundations of Computer Science* (FOCS '87), pages 427-437.

[Ped91a] Torben P. Pedersen: Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In *Proceedings of Advances in Cryptology* - CRYPTO 1991, volume 576 of LNCS, pages 129-140. Springer-Verlag, 1991.

[Ped91b] Torben P. Pedersen: A Threshold Cryptosystem without a Trusted Party (Extended Abstract). In *Proceedings of Advances in Cryptology* - EUROCRYPT 1991, volume 547 of LNCS, pages 522-526. Springer-Verlag,1991.

[BKW97] Johannes Blomer, Richard M. Karp, Emo Welzl: The rank of sparse random matrices over finite fields. *Random Structures and Algorithms*, volume 10 number 4, pages 407-419. (1997)

[SG98] Victor Shoup, Rosario Gennaro: Securing Threshold Cryptosystems against Chosen Ciphertext Attack. In *Proceedings of Advances in Cryptology* - EUROCRYPT 1998, volume 1403 of LNCS, pages 1-16. Springer-Verlag, 1998.

[GJKR99] Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, Tal Rabin: Secure Distributed Key Generation for Discrete-Log Based Cryptosystems. In

Proceedings of Advances in Cryptology - EUROCRYPT 1999, volume 1592 of LNCS, pages 295-310. Springer-Verlag, 1999.

[SZT02] Dawn Xiaodong Song, J. D. Tygar, David Zuckerman: Expander Graphs for Digital Stream Authentication and Robust Overlay Networks. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy.* Pages 258-269.

[GJKR03] R. Gennaro, S. Jarecki, H. Krawczyk, T. Rabin. Revisiting the distributed key generation for discrete-log based cryptosystems. 2003.

[Ngo04] Hung Q. Ngo. Combinatorial and Graph Algorithms. (Lecture 1: Matchings on bipartite graphs)

[CS04] John F. Canny, Stephen Sorkin: Practical Large-Scale Distributed Key Generation. In *Proceedings of Advances in Cryptology* - EUROCRYPT 2004, volume 3027 of LNCS, pages 138-152. Springer-Verlag, 2004.