國立政治大學資訊科學系

Department of Computer Science

National Chengchi University

碩士論文

Master's Thesis

具隱私保護功能之兩方相等性驗證機制之提案

Two-party Equality Test with Privacy Protection

研 究 生: 邱士峰

指導教授：左瑞麟

中華民國一〇一年一月

January 2012

具隱私保護功能之兩方相等性驗證機制之提案

Two-party Equality Test with Privacy Protection

研 究 生：邱士峰　　Student：Shih-Fong Ciou

指導教授：左瑞麟　　Advisor：Ray-Lin Tso

國立政治大學

資訊科學系

碩士論文

A Thesis

submitted to Department of Computer Science

National Chengchi University

in partial fulfillment of the Requirements

for the degree of

Master

in

Computer Science

中華民國一〇一年一月

January 2012

致謝

在研究所這兩年的日子裡，有難過也有歡笑；有困惑也有收穫，但很高興最後論文能順利完成。首先我要感謝指導教授左瑞麟老師的細心教導，在嚴謹的訓練和彼此的討論下，教會我如何從文獻中找到問題並嘗試提出更好的解決方式。過程中雖然辛苦，但也培養我面對問題及解決問題的能力，更從中得到一些知識和充實感。

論文撰寫期間，感謝左瑞麟老師及的督促與指導，帶領不善文筆的我一步步完成碩士論文。並感謝口試委員羅乃維教授、吳宗杉教授給予我許多寶貴的意見，使這篇論文更加明確及完整。

在研究所生活裡相當精彩，首先，感謝政大資科的學長、同學致諺、學弟，在我研究之餘還能夠從事不少的球類運動及電玩遊戲，讓我在快樂的學習環境下，減輕不少生活中無形的壓力，感謝學長們在閒暇之時，帶我遊玩蠻多地方，讓我不是一昧的待在實驗室。最後要感謝我的父母，在我的求學過程中無私的奉獻，謝謝你們。

具隱私保護功能之兩方相等性驗證機制之提案

摘要

本研究的研究目的是比較雙方秘密數值是否相等，而在以往的安全多方計算的研究，通常雙方的秘密數值經過協定之後，一個為告知方，另外一個為被告知方，由告知方通知計算後之結果，而被告知方只能相信此訊息。如果藉由半誠實的第三方可解決上述問題並減少計算量，但找到可以信任的第三方是比較不容易的。

基於以上問題，本研究提出一新的秘密計算協定，在此協定下參與的雙方(告知方、被告知方)可以算出彼此所擁有的秘密是否相同。如果不同，此協定不會洩漏任何秘密值的資訊。本方案亦提供驗證機制，讓被告知方能驗證告知方是否屬實。

Two-party Equality Test with Privacy Protection

Abstract

The purpose of this study is to compare the equality of two secret values. Secure multiparty computation in the previous study, usually through the protocol the two sides, the one is announcer and the other one be told. The one be told by the announcer who notified the results of verification, and the one be told only can believe that the message. Through the semi-honest party can solve by the above problems and reduce the computation required, but you can find a trusted third party is not easy.

Based on the above problems, this study proposed in the framework of both the secret of a new calculation of protocol, in this protocol the two parties (the one is announcer, the other one be told) can calculate each have a secret are equal or not. If different, this protocol does not leak any information about the secret value.

目錄

圖目錄

表目錄

第一章　緒論

1.1 研究背景

密碼學一直以來都是電腦網路、資訊安全等應用上重要的一環，而密碼學也促進了在網路安全上所使用的技術，例如存取資料方面的隱密性，現在的日常生活中也可見其相關的應用，例如 ATM 的晶片卡……等。

隨著網路世界的發達與普及，許多使用者開始運用網路來進行交流，而在交流的過程中是否伴隨著資訊的洩漏呢？這是我們所要注意的，畢竟沒人想把自己私密的訊息公開給大眾知道，而私密的訊息經過加密過後可達保護訊息的作用，但經過加密後的訊息通常要再經過解密後才能做另外的運算或者其他動作，所以我們就需要考慮在加密且不會洩漏訊息本身的前提下，仍可以經過某些協議過程之後達到想要的結果，而能達到此方法的就是安全多方計算(secure multi-party computation)，這也成為我們研究的重點。

安全多方計算是密碼學中的一環，安全多方計算是指：

1. 兩方或更多參與者基於他們各自私密資訊輸入的計算。

2. 參與者都不希望其他參與者知道自己輸入的秘密資訊。

3.參與者只能得到最終結果，除了最終結果以外不會在得到任何而外資訊。

在一個互不信任的多使用者網路中，利用安全多方計算，各使用者能夠通過網路來協同完成可靠的計算任務，同時又能達到資訊安全跟隱私保密的功能。

安全多方計算問題可以描述為：n 個參與者$\{p_1, p_2, \dots, p_n\}$，此 n 個參與者各自擁有自己的私有資訊，分別為$\{X_1, X_2, \dots, X_n\}$，希望經由一個公開的函數f，在不公開 n 筆私密訊息($\{X_1, X_2, \dots, X_n\}$)的情況下做相關運算，運算結束後得到想要的結果。

舉例來說，安全多方計算協議要解決的問題可以描述如下：兩位參與者(A 和 B)希望共同計算某個約定的函數(如：$f(a,b)=a+b$ 或者 $f(a,b)=a<b$...等)，每個參與者提供函數的一個輸入(a 和 b)，出於安全考慮，要求參與者提供的輸入對其他人保密。假如存在安全可信任的第三方，則所要解決的問題可以輕易地得到解決：只需各參與者將各自的輸入交給第三方，由第三方計算出函數值，再將函數值公佈給各參與者。但現實中很難找到這樣安全可信任的第三方，從而安全多方計算協議的研究應運而生，目前安全多方計算已得到許多學者的研究，其在密碼學上的地位也日益重要。

安全多方計算的研究議題，主要起源於 1982 年 Yao[21]所發表的百萬富翁問題，安全多方計算已經發展成為現代密碼學中一個非常活躍的研究領域，而後 Goldreich[12]對安全多方計算有比較細緻系統的論述。

1.2 研究動機與目的

在現在資訊發展迅速的情況之下，安全多方計算漸漸重要，我們運用在當某些數據需要經過計算、比較，但又不想把這些屬於秘密的數據公開，在這種情況之下，在經過安全多方計算協定(multi-party computation protocol)的過程之後，就能把這些數據做運算且不會洩漏不想公開的秘密資訊。如：點、線、面方面，當參與者(A 和 B)需要計算某個點(A 所擁有的私密資訊)是否包含在某個面(B 所擁有的私密資訊)之內，且在計算的過程中不會洩漏彼此私密的訊息，這樣的情況下就需要安全多方計算來做運算來達到目的。

安全多方計算有很強的應用背景，特別是在電子資訊的時代，舉例來說，像現在很多人用的 facebook，其實很多個人資訊很容易就被陌生人看得一清二楚，所以需要把不想給陌生人看的資訊隱藏起來。由下圖 1 來說明，此使用者(A)把資訊全部隱藏起來，由圖可見當一個還沒加入好友的人(B)並沒有多餘的訊息可以確認，所以 B 並沒有辦法肯定 A 是認識的人。假如 B 想加入 A 為好友，但 A 是不認識的人但名字一樣，一旦加入之後 B 的個人資料就被全都被 A 知道了，或者 B 傳送訊息給 A，A 卻欺騙 B，而讓 B 資料被 A 所知。為了確保自己的資料安全，在這樣的情況下，秘密比較的功能就能派上用場。系統可以設定輸入一些資料來比對，如生日、年齡、血型……等，然後把資料傳到 sever 端來做比對，假如 sever 端告訴 B 比對正確，那也就代表 A 是 B 認識的人，這樣 B 才能安心把 A 加入好友，假如失敗的話，也是無法從中得到任何資訊，這樣也能保證被查的人的資料安全。

除此之外，安全多方計算還可應用在，例如網路上電子投標，網路上商業相關、電子選舉計票、比較年齡等問題。如何設計高效、安全的保密比較雙方或多方數值大小(或相等)的協定是一個極具挑戰的問題，也是密碼學中的一項重要研究任務。

圖 1 Facebook 資訊圖

安全多方計算中的一個特例就是安全雙方計算(secure two-party computation)。亦即，參與者只有兩個人，本研究就著重在於雙方參與者經過協定之後能比較數值是否相等，在不相等的情況雙方無法從中得到任何一點秘密的訊息。只有在雙方相等的情況才能得知彼此的數值，而在目前我們所看到的協定之下，透過半誠實的第三方[6][14][19]所實施的安全多方計算的協定，雖然讓雙方參與者運算的情況可以改善且步驟可能比較不複雜，但要找到一位讓雙方參與者都信任的第三方，並不是那麼容易就能找到的，只要此第三者偏向某一方，而另一方的秘密資訊就會因此容易被揭露出來。由於透過第三方會有上述問題，所以本研究將以雙方來進行協定，而不透過第三方來達到秘密計算的目的。

1.3 研究貢獻

在現有的雙方安全多方計算[2][6][10][21]我們可以看到，在協定下雙方做秘密計算之後，通常由一位告知者，另外一位為被告知者，而被告知者要如何確定告知者是把協定過後的值正確的公佈呢？我們研究的貢獻在於：

1. 經由本研究的協定，不需要透過第三方我們可以精確的計算出雙方數值是相等，不相等的情況也不會洩露出彼此的秘密資訊。
2. 在本研究的協定，讓以往雙方安全多方計算的被告知者也能驗證告知者告知的結果是否正確。
3. 在本研究的協定，計算時間上雙方都不到一秒(不包含傳輸時間和亂數選取時間)，所以在計算方面負擔並不會太大。

1.4 論文架構

在本論文中，共分為六個章節來做探討，各章節內容架構大致如下：

- 第一章為緒論，以本篇論文的研究背景與研究動機與目的作一簡單的說明。
- 第二章為相關研究，對於我們文章後續將用到的一些技術如：同態公開金鑰加密，同時對於一些密碼學的基本知識，和本研究相關的文獻，作一簡單的介紹。
- 第三章為介紹我們基本想法的協定，經過此基本協定參與的雙方(告知方和被告知方)數值是否相等的結果只有告知方得知，最後由告知方告知結果。
- 第四章為在第三章的基本協定下做改良，提出一個完整的協定，經過此協定能知道參與的雙方(告知方和被告知方)數值是否相等，且讓被告知方也能達到驗證的功能。
- 第五章利用我們的協定製作簡單的程式，來檢測我們協定的運算時間。
- 第六章為結論與未來展望。

第二章　相關研究

在此章節首先我們會介紹一些密碼學的基礎知識，還有在本研究的協定所需使用到的一些加密演算法，最後介紹一些跟本研究相關的文獻。

2.1 近代密碼學簡介

密碼學主要研究是如何秘密地傳遞訊息，近代以前的密碼學純粹專指加密演算法，將可理解的信息（明文”plaintext”）轉換成難以理解的信息（密文”ciphertext”）[20]，解密演算法則是其相反的過程，由密文轉換回明文。一個用來加解密的系統包含了這兩種演算法，而我們就稱其為一個密碼系統(Cryptosystem)。架構圖如下圖:

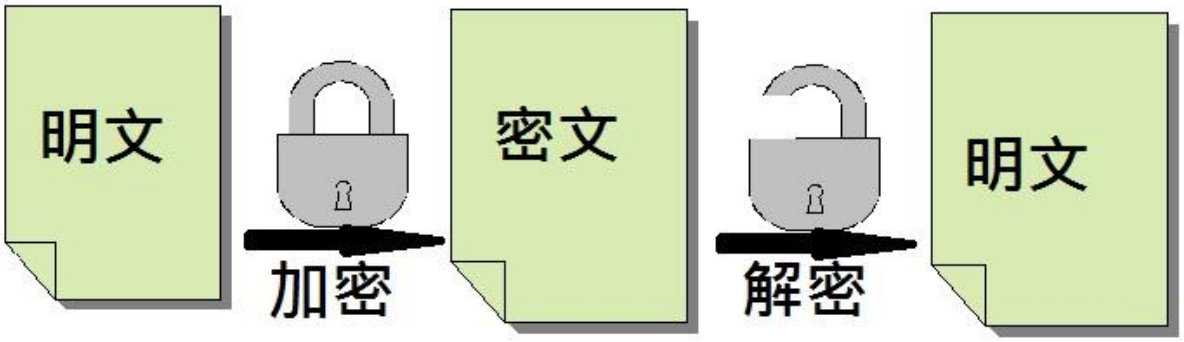

圖 2 密碼系統

而密碼系統主要可分為兩大類，分別為「對稱式金鑰密碼系統」以及「非對稱金鑰密碼系統」。

2.1.1 對稱式金鑰密碼系統(Symmetric Key Cryptosystem)

在此密碼系統中，發送者與接收者在彼此的通訊開始前，必須先決定一對雙方都持

有的共同金鑰。確定後，發送者使用這把(共同)金鑰對明文加密(Encryption)：$E_K(m) = C$，E 為加密演算法，K 為金鑰，m 為明文，C 為密文；接收者接收到密文 C 後也利用同一把金鑰做解密(Decryption)：$D_K(C) = m$，D 為解密演算法。

由此可見，在此密碼系統中，最主要就是保證這把雙方共享的金鑰不會有洩漏的疑慮。而採用這種方式的密碼系統，有 DES 密碼系統[3]、AES 密碼系統[4]以這兩種比較為人所知。

在這種密碼系統中，有幾個問題是我們著重且須要注意的，為：

1. 密鑰 K 要怎麼安全的傳送給接收者。
2. 如果系統中有 N 個使用者，兩兩之間需要通訊的話，那麼就需要產生C_2^N對金鑰，所以每個人所持有的金鑰數量就會相當可觀。

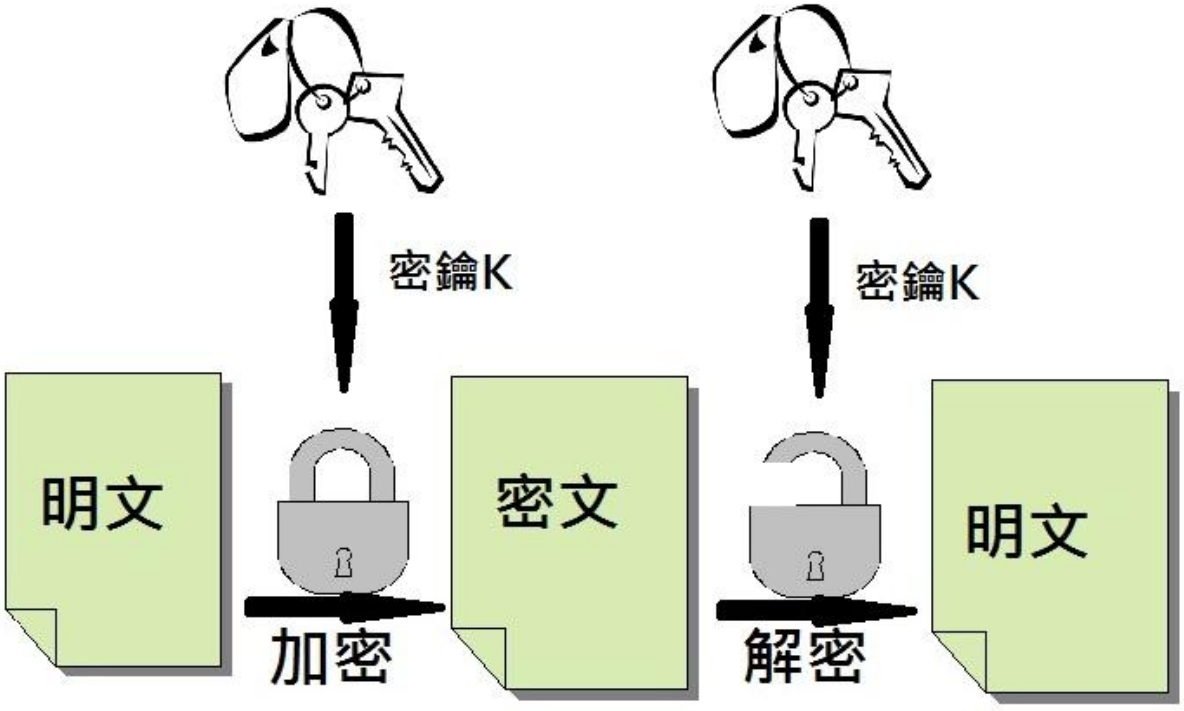

圖 3 對稱式金鑰密碼系統

2.1.2 非對稱式金鑰密碼系統(Asymmetric Key Cryptosystem)

此密碼系統又稱為「公開金鑰密碼系統」，跟對稱式金鑰密碼系統最大的差異就是加密和解密使用不同的金鑰。

1976 年，Diffie 和 Hellman 這兩位學者提出了公開金鑰密碼系統的概念[7]。此密碼系統的金鑰是成對的，兩個金鑰是數學相關，使用「公開金鑰」加密，要以「私密金鑰」才能解密，且無法以公開金鑰推算私密金鑰。比較著名的系統有 RSA[18]、ElGamal[9]……等。此密碼系統在數位簽章(Digital Signature)上也有很多的利用。

數位簽章依附於電子文件上，用以辨識以及確認電子文件簽屬者之身分和文件的正確性，數位簽章為兩把不同的鑰匙(非對稱式)，一把用於簽名，另一把用於驗證，簽名的時候用私鑰，驗證簽名的時候則用公鑰，本研究重點並不在數位簽章，因此有興趣可參考相關文獻[9][16][18]。

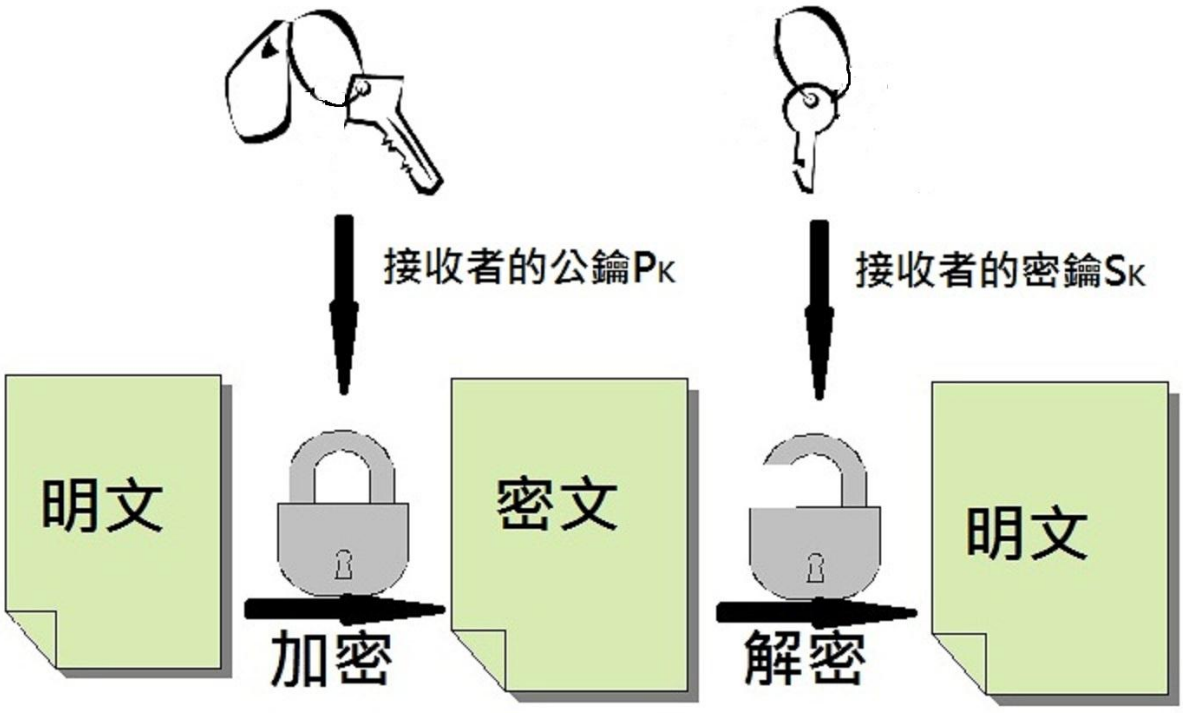

圖 4 非對稱式金鑰密碼系統

2.2 同態公開金鑰加密(Homomorphic Encryption)

同態是從一個代數系統(Algebraic system)到同類代數系統(例如群(Group)、環(Ring)或者向量空間(Vector space))的對應關係，運算過程中保持所有相關的結構不變，也就是說單位元素(Unit element)、反元素(Inverse element)和二元運算(Binary compute)的屬性不變。舉例來說：兩自然數 *a* 和 *b*，若函數 *f(x)*滿足 *f(a+b)=f(a)+f(b)*，則函數 *f(x)*具有同態的性質。

同態加密系統是一些可以在密文間做代數運算的加密演算法，如將密文$c_1 = E(m_1)$和密文$c_2 = E(m_2)$做運算得到$c = c_1 \odot c_2$等同於$c = E(m_1 \oplus m_2)$ (c 為密文，m 為明文，E 為有同態性質的加密演算法，$\odot$和$\oplus$為某種運算)。此方法可節省運算成本(不需要多餘的加解密)，還可以保障運算過程中的訊息安全。

2.2.1 加法同態

加法同態為兩的密文做某種運算，會等同於明文做加法運算再加密，如下：

$$c_1 = E(m_1)，c_2 = E(m_2)$$

$$c = c_1 \oplus c_2 = E(m_1) \oplus E(m_2) = E(m_1 + m_2)$$

舉例：在$E(m) = km$的情況，$E(m_1) + E(m_2) = E(m_1 + m_2)$，或者$E(m) = e^m$，這樣$E(m_1) * E(m_2) = E(m_1 + m_2)$，都能達到加法同態。在文獻 Paillier [17] Naccache-stern [15] Boneh-Goh-Nissim [1]都能看到加法同態。

在此以 Paillier 的加密系統法來說明，此系統由三個演算法：Key Generation、Encryption 、Decryption 組成

Key Generation(金鑰生成演算法)：假設密文接收者為 Alice，則 Alice 利用 Key Generation 產生自己的私鑰。其方法如下

1. 選擇兩個大質數 p 和 q

2. 計算 $N=p*q$ 和$\lambda = lcm(p-1, q-1)$，其中 *lcm*(Least Common Multiple)代表 $p-1, q-1$最小公倍數。

3. 選擇亂數$g \in Z_{N^2}^*$，$Z_{N^2}^*$是比N^2小且與N^2互質的元素且包含單位元素 1 的集合。

4. 函數 $L(u)=(u-1)/N$

5. 設g的序為l，確保$N|l$

6. Alice 的公鑰 $pk=(N, g)$

7. Alice 的私鑰 $sk=(N, \lambda)$

Encryption(加密演算法)：假設 Bob 欲利用 Paillier encryption 加密訊息 *m* 給 Alice，則 Bob 利用加密演算法執行以下步驟

1. 欲加密訊息 $m \in Z$

2. 選擇亂數 $r \in Z^*$

3. 計算密文 $C=E(m)=g^m * r^N \ mod \ N^2$

Decryption(解密演算法)：Alice 收到密文 C 後利用解密演算法做以下步驟解密

1. 密文 $C \in Z_{N^2}^*$

2. $m=D(C)=(\frac{L(C^\lambda \ mod \ N^2)}{L(g^\lambda \ mod \ N^2)}) \ mod \ N$

Paillier 加法同態：

假設 Bob 把m_1和m_2分別加密起來，得到 $E(m_1)=g^{m_1} * r_1{}^N$和 $E(m_2)=g^{m_2} * r_2{}^N$，把 $E(m_1)$ 和 $E(m_2)$傳送到資料庫，資料庫在不知道明文的情況下，對兩個密文可做同態運算，如下：

$$C=E(m_1)*E(m_2)=(g^{m_1} * r_1{}^N)*(g^{m_2} * r_2{}^N)=g^{m_1+m_2} * (r_1 r_2)^N=E(m_1+m_2)$$

Alice 可以從資料庫得到 C，但得不到單獨的訊息，所以 Alice 解密 C 的時候就能得到 $m_1 + m_2$

2.2.2 乘法同態

乘法同態為兩的密文做某種運算，會等同於明文做乘法運算再加密，如下：

$$c = E(m_1) \otimes E(m_2) = E(m_1 * m_2)$$

舉例：在$E(m) = m^e$這樣$E(m_1) * E(m_2) = \mathrm{m_1}^{\mathrm{e}} * \mathrm{m_2}^{\mathrm{e}} = E(m_1 * m_2)$。而 RSA[18]、ElGamal[9]、Boneh–Goh–Nissim[1]都是乘法同態。

我們舉 RSA 來說明：

Key Generation：假設密文接收者為 Alice，則 Alice 利用 Key Generation 產生自己的公私鑰。其方法如下

步驟 1. 隨意選擇兩個大的質數p和q，$p \neq q$，計算$N = pq$。

步驟 2. 根據尤拉函數，不大於N且與N互質的整數個數為$(p-1)(q-1)$

步驟 3. 選擇一個整數e與$(p-1)(q-1)$互質，並且e小於$(p-1)(q-1)$

步驟 4. 用以下這個公式計算d：$d * e \equiv 1\ (mod\ (p-1)(q-1))$

步驟 5. 將p和q的記錄銷毀。

(N, e)是公鑰，(N, d)是私鑰。將公鑰(N, e)公開，而私鑰(N, d)秘密保存。

Encryption：假設 Bob 欲利用 RSA encryption 加密訊息 m 給 Alice，Bob 利用 Alice 的 $pk=(N, e)$計算密文 C：

$$C = m^e\ mod\ N$$

Decryption：Alice 收到密文 C 後利用 $sk=(N,d)$解密 C 得

$$m = C^d\ mod\ N$$

分析與討論：

$c^d \equiv m^{e*d}(mod\ N)$，$\mathrm{ed} \equiv 1\ (\mathrm{mod}\ \mathrm{p}-1)$和$\mathrm{ed} \equiv 1\ (\mathrm{mod}\ \mathrm{q}-1)$。

由費馬小定理可證明（因為p和q是質數）

$m^{e*d} \equiv m(mod\ p\ )$

$m^{e*d} \equiv m(mod\ q)$

因為p和q是*不同*的質數，所以p和q互質

$m^{e*d} \equiv m(mod\ p * q)$

RSA 乘法同態：

Bob 產生密文 $c_1 = {m_1}^e\ (mod\ N)$ 和密文 $c_2 = {m_2}^e\ (mod\ N)$，傳送到資料庫，資料庫在不知道明文的情況下，對兩個密文可做同態運算，如下：

$$C = E(m_1) * E(m_2) = {m_1}^e * {m_2}^e = (m_1 * m_2)^e (mod\ N)$$

Alice 可以從資料庫得到 C，但得不到單獨的訊息，所以 Alice 解密 C 的時候就能得到 $m_1 * m_2$。

2.3 可換式加密(Commutative Encryption)

一個訊息 m 被加密兩次以上，而其加密演算法滿足下列：

1. 假設(pk_1, sk_1)為 user 1 的公私鑰，(pk_2, sk_2)為 user 2 的公私鑰，則$E_{pk_1}\left(E_{pk_2}(m)\right) = E_{pk_2}\left(E_{pk_1}(m)\right)$。

2. 加密鑰匙pk_i所對應的解密鑰匙sk_i在多項式時間可計算。

3. 利用pk_1或pk_2加密所產生的密文，其值域相同。

滿足上述三項性質的加密演算法稱為可換式加密(Commutative Encryption)。簡言之，在 Commutative Encryption 中無論先用哪個加密鑰匙加密跟密文結果並不影響，而誰先解開此密文並不會影響其明文的結果(可見條件 1)。

舉例來說，ElGamal[9]原先的演算法並不符合 Commutative Encryption，所以我們假設一些條件來達成，文章後會再說明，在此先介紹 ElGamal 加密，其由三個演算法 Key Generation、Encryption、Decryption 所組成。

Key Generation(密鑰產生) 假設密文接收者為 Alice，則 Alice 利用 Key Generation 產生自己的私鑰。其方法如下

步驟 1. 產生乘法群G和生成元(generator)g，其中q為G這個群中的元素個數。

步驟 2. 從0到$q-1$之中選擇一個亂數 x

步驟 3. 計算$h = g^x \ mod\ q$

步驟 4. 公開 (G, q, g, h) 做為公鑰，x為私鑰。

Encryption(加密步驟)：假設 Bob 欲利用 ElGamal encryption 加密訊息 m 給 Alice，Bob 利用加密演算法執行以下步驟

明文 m

公鑰 (G, q, g, h)

步驟 1.　　從0到$q-1$之中選擇一個亂數y使得$C_1 = \mathrm{g}^y \ mod\, q$

步驟 2.　　計算共享秘密$s = h^y \ mod\, q$。 s又稱為暫時性的鑰匙，用來跟每段明文做運算。

步驟 3.　　將明文m轉換為G這個群中的元素m'。

步驟 4.　　計算$C_2 = m' * s \ mod\, q$。

步驟 5.　　(C_1, C_2)為密文 C。

Decryption(解密步驟)：Alice 收到密文 C 後利用解密演算法做以下步驟解密

私鑰 x

步驟 1.　　計算共享秘密$s = {C_1}^x \ mod\, q$

步驟 2.　　$m' = C_2 * s^{-1} = m' * s * s^{-1} = m' \ mod\, q$

前面有提到，為了達到 Commutative Encryption，我們假設由系統來生成兩個 key(其G, q, g都相同)來給 user 1 和 user 2。如下：

以下運算都在 $mod\, q$ 的環境下進行。

$h_1 = g^{x_1}$ 、$h_2 = g^{x_2}$

分別為：

$C_1 = g^{y_1}$ 、$C_2 = g^{y_2}$

$s_1 = h^{y_1}$ 、$s_2 = h^{y_2}$

我們可以看到：

$E_1(m) = \ m{*}s_1$ ，$E_2(m) = m * s_2$

所以：

$$E_1\big(E_2(m)\big) = m * s_2 * s_1 = E_2\big(E_1(m)\big)$$

也可以看出不管先把哪個加密解開並不會影響其值。

2.4 語意安全(Semantic security)

語意安全是一個廣泛使用在證明一個非對稱金鑰加密演算法的安全性，一個非對稱金鑰加密演算法若具有語意安全，則攻擊者無法從密文得到有關明文的訊息

語義安全的概念最早是由 Goldwasser 和 Micali[13]在 1982 年所提出，它具有抵抗選擇明文攻擊，證明時會設計一個遊戲，假如贏得遊戲，代表具有能破解語意安全的能力。

攻擊者 B 有加密方法E與公開金鑰 pk ，首先，參與者 A 產生兩個相等長度的訊息m_0和m_1，參與者 A 選取一個m_b，$b\in\{0,1\}$，利用加密方法E加密 $C=E_{pk}(m_b)$，加密後將$\{m_0, m_1, C\}$傳送給攻擊者 B，攻擊者 B 猜中 C 是以m_0或m_1加密的機率為 $1/2+\varepsilon$, 其中若值 ε 為可忽略的(negligible), 則稱此加密法具有語意安全。

舉例說明，再拿 ElGamal[9]來說明：

假設 A 方生成m_0和m_1，也把這個資訊告訴 B 方，A 方把一個訊息加密C=$E_{pk}(m_b)$，$b\in\{0,1\}$。

我們可以看到 ElGamal 的加密方法$h = g^x$，$c_1 = g^y$，$s = h^y$，$c_2 = m * s \mod q$。Alice 利用 ElGamal 加密得到密文 $C=(c_1, c_2)$。

我們假設 A 方所加密的訊息是m_1，公鑰為(G, q, g, h)，B 方知道c_1和c_2的值，但由於 y 是由 A 亂數選取，是 B 方不知道的且沒辦法從已知的訊息計算出來，所以 s 這個數值也是 B 方所不知的，B 方推導$c_2 = m * s$，我們可看到 s 和 m 對於 B 方都是未知的，所以當 B 方猜測一個 m 的值，也會對應一個 s 值，滿足$c_2 = m * s$，所以 B 方並沒有辦法判斷C是由m_0和m_1哪一個訊息所加密而成的，所以就能達到具有語意安全。

2.5 雙重同態加密(Doubly-homomorphic Encryption)

一個加密的方案包含了加法同態跟乘法同態，我們就可以稱為 Doubly-homomorphic encryption。如下：

給定$E(x)$和$E(y)$，在不知道 x 和 y 的情況之下，任何人都能計算出$E(x*y)$和$E(x+y)$。

在現今的文獻中，從 Boneh–Goh–Nissim[1]，2005 年所發表的文獻中就包含 Doubly-homomorphic encryption，但整體架構只能做一次乘法同態，所以並不是非常的完整，直到最近 Gentry[11]，2009 年所發表的文獻中，利用對稱式加密來達到 Doubly-homomorphic encryption，讓此方面的研究更為精進。

接著介紹 Gentry 文獻中的加密方法。在此以文獻中對稱式加密為例說明：

假設密文接收者為 Alice，密文傳送者為 Bob

Shared secret key: 由 Alice 和 Bob 產生共享金鑰(Shared secret key)。其方法如下

選一大亂數 p 且 p 為奇數。

由於利用對稱式加密，知道此金鑰就能解密，系統會將鑰匙給與 Alice 和 Bob。

加密 m：Bob 計算產生密文

加密的訊息 $m\in\{0,1\}$。

Bob 選擇一小亂數 $r'\in Z^*$，跟一大亂數 $q\in Z^*$，$r'\ll q$。

Bob 計算 $c = m + 2r' + pq$ 。

假如 Bob 有 n 個訊息$\{m_1, m_2, \dots\dots, m_n\}$要做運算，所對應的密文 $C=\{c_1, c_2, \dots\dots, c_n\}$。

運算：由 Bob 將密文 C 傳送到運算端做運算

假如想得到乘法的結果，運算端就做乘法$C_1 = \{c_1 * c_2 * \dots\dots * c_n\}$

假如想得到加法的結果，運算端就做加法$C_2 = \{c_1 + c_2 + \dots\dots + c_n\}$

解密：Alice 從運算端收到密文C_1或C_2做以下步驟解密

輸出 $m_1 * m_2 * \ldots\ldots * m_n = (C_1 \mod p) \mod 2$

$m_1 + m_2 + \ldots\ldots + m_n = (C_2 \mod p) \mod 2$

Alice 經過解密之後就能得到想要的結果。

此演算法如何達到 Doubly-homomorphic encryption 呢？

我們知道需要滿足能計算$E(x * y)$和$E(x + y)$。

在$E(x * y)$，也就是乘法同態的部分：

我們把 x 和 y 加密如下：

$E(x) = x + 2r_1' + pq_1$

$E(y) = y + 2r_2' + pq_2$

而我們把$x * y$加密如下：

$E(x * y) = x * y + 2r_3' + pq_3$

把$E(x)$和$E(y)$相乘，也就是檢驗是否有乘法同態：

$$E(x) * E(y) = x * y + x * 2r_2' + x * pq_2 + 2r_1' * y + 2r_1' * 2r_2' + 2r_1' * pq_2 + pq_1 * y + pq_1 * 2r_2 + pq_1 * pq_2$$

經過整理：

$$E(x) * E(y) = x * y + 2(x * r_2' + r_1' * y + r_1' * 2r_2' + r_1' * pq_2 + pq_1 * r_2') + p(x * q_2 + q_1 * y + q_1 * pq_2)$$

將$E(x * y)$和$E(x) * E(y)$解密，也就是 $mod\ p$ 再 $mod\ 2$ 所得的結果都是$x * y$代表滿足了

乘法同態。

在$E(x+y)$也就是加法同態的部分：

而我們把$x+y$加密如下：

$$E(x+y)=(x+y)+2r_4{}'+pq_4$$

將$E(x)$和$E(y)$相加，也就是檢驗是否有加法同態：

$$E(x)+E(y)=\ x+2r_1{}'+pq_1+\ y+2r_2{}'+pq_2$$

經過整理：

$$E(x)+E(y)=(x+y)+2(r_1{}'+r_2{}')+p(q_1+q_2)$$

把$E(x+y)$和$E(x)+E(y)$解密，一樣是 *mod p* 再 *mod 2* 所得的結果都是$x+y$這也代表滿足了加法同態。

由此舉例可知，Gentry 的演算法滿足能計算出$E(x*y)$和$E(x+y)$的同態性質，所以我們可說此演算法有 Doubly-homomorphic encryption 的性質存在。

2.6 安全秘密計算(在雙方架構下)

我們可以從 1982 年 Yao[21]所發表的文獻，被稱之為百萬富翁問題，百萬富翁問題是一種安全多方計算的問題的特例。假設如下：

有兩個百萬富翁，Alice 和 Bob 都宣稱自己很有錢，所以想要知道誰比較有錢，但有不想透漏彼此真正的財產給對方知道，如何可以不用透露彼此真實的財產就能知道結果呢?使用公開金鑰密碼系統，經由此協定就能解決此問題。

2.6.1 Yao 的協定

假設 i 為 Alice 的秘密資訊，j 為 Bob 的秘密資訊，Alice 和 Bob 想比較彼此秘密訊息的大小。

1. Bob 挑一個任意 N-bit 整數 x，計算 $k=E_{pk}(x)$，

   E_{pk}為 Alice 以 RSA(參閱第 2.2.2 節)所產生的加密演算法。

2. Bob 傳給 Alice：*k-j+1*

3. Alice 對 m 個數值做解密$Y_u = D_{sk}(k-j+u), u = 1,2,\dots,m$。

   D_{sk}為 Alice 以 RSA(參閱第 2.2.2 節)所產生的解密演算法。

4. Alice 挑一個任意 *N/2-bit* 質數 p

$$Z_u = \begin{cases} y_u \bmod p, & u = 1,2,\dots,i \\ y_u \bmod p + 1, & u = i+1, i+2, \dots, m \end{cases}$$

(慎選 p 已使Z_u相差至少 2)

5. Alice 將 p 與Z_u傳送給 Bob。

6. Bob 計算 *G=x mod p*

$$Z_j = G \rightarrow i \geq j$$
$$Z_j \neq G \rightarrow i < j$$

7. Bob 告訴 Alice 結果。

以下舉個例子以便能理解：

假設，Alice 有 *i=5* 百萬，Bob 有 *j=6* 百萬 *1<i,j<m=10*，以 RSA 做為公鑰加密系統，公鑰(e,N)=(5,119)，私鑰(d,N)=(77,119)

1. Bob 挑一個任意 N-bit 整數 *x=234(8bits)*，以 Alice 的公鑰加密為 $k=234^5 \ mod\ 119=47$。
2. Bob 傳給 Alice：*k-j+1=42*。
3. Alice 對 *m* 個數值做解密 $Y_u = D_{sk}(k-j+u), u = 1,2,\dots,m$。

表格 1 EX1

u	k-j+u	RSA	y_u
1	42	42^77 mod 119	77
2	43	43^77 mod 119	8
3	44	44^77 mod 119	11
4	45	45^77 mod 119	75
5	46	46^77 mod 119	65
6	47	47^77 mod 119	115
7	48	48^77 mod 119	90
8	49	49^77 mod 119	70
9	50	50^77 mod 119	50
10	51	51^77 mod 119	102

4. Alice 挑一個任意 *N/2-bit* 質數 *p=11(4bits)*。

$$Z_u = \begin{cases} y_u \bmod p, & u = 1,2,\dots,i \\ y_u \bmod p + 1, & u = i+1, i+2, \dots, m \end{cases}$$

(慎選 p 已使Z_u相差至少 2)

表格 2 EX2

u	k-j+u	RSA	y_u	Z_u
1	69	69^77 mod 119	77	0
2	70	70^77 mod 119	8	8
3	71	71^77 mod 119	11	0
4	72	72^77 mod 119	75	9
5	73	73^77 mod 119	65	10
6	74	74^77 mod 119	115	6
7	75	75^77 mod 119	90	3
8	76	76^77 mod 119	70	5
9	77	77^77 mod 119	50	7
10	78	78^77 mod 119	102	4

5. Alice 將 p 與Z_u傳送給 Bob。

表格 3 EX3

p				
11				
Z_1	Z_2	Z_3	Z_4	Z_5
0	8	0	9	10
$Z_6 + 1$	$Z_7 + 1$	$Z_8 + 1$	$Z_9 + 1$	$Z_{10} + 1$
6	3	5	7	4

6. Bob 計算 *G=x mod p =5*

$$Z_j = G \rightarrow i \geq j$$
$$Z_j \neq G \rightarrow i < j$$
$$Z_6 = 6 \neq G = 5 \rightarrow i < j$$

7. Bob 告訴 Alice 結果。

藉由這幾個步驟這樣就能得知結果，但假如是 *i=6，j=5* 或者 *i=6，j=6* 這樣的情況下，我們必須再做一次所有的步驟，這次須交換執行，才能得到最後結果。

問題點：

此方法使我們在秘密計算有重大的發展，我們可以在不洩漏自己秘密的資訊，來達到比較雙方數值的大小，但是效率有點不好，如果 *i* 不小於 *j* 這樣的話就需要跑兩次協定。另一個問題就是，就是有告知方和被告知方的情況，被告知方只能從告知方得到結果，被告知方沒辦法驗證，只能相信告知方告知的結果，這是此方法的兩個問題。

在雙方秘密計算[2][6]的情況下我們可以看到，基本上都有被告知方只能從告知方得到結果的問題。

圖 5 雙方通訊示意圖

2.7 安全秘密計算(藉由半誠實的第三方)

在此利用第三方來比較雙方數值是否相等，和前面 2.6 所介紹的結果是不同的。

2.7.1 半誠實的第三方(Semi-trust Third Party)

何謂半誠實的第三方?假如是完全誠實的第三方，只要雙方參與者把沒有加密的數值直接告訴第三方，如圖 6，或者雙方參與者用第三方的公鑰加密數值傳給誠實的第三方，由第三方解密、比較過後，直接告訴參與的雙方比較結果這樣就可以了，當然這是在完美的情況之下。半誠實的第三方則是指，第三方會照著協定的規則方法進行通訊和計算，途中不會得到甚麼額外秘密資訊，但如果協定本身有任何缺失，第三方也能從中攻破得到其秘密的資訊，如圖 7。

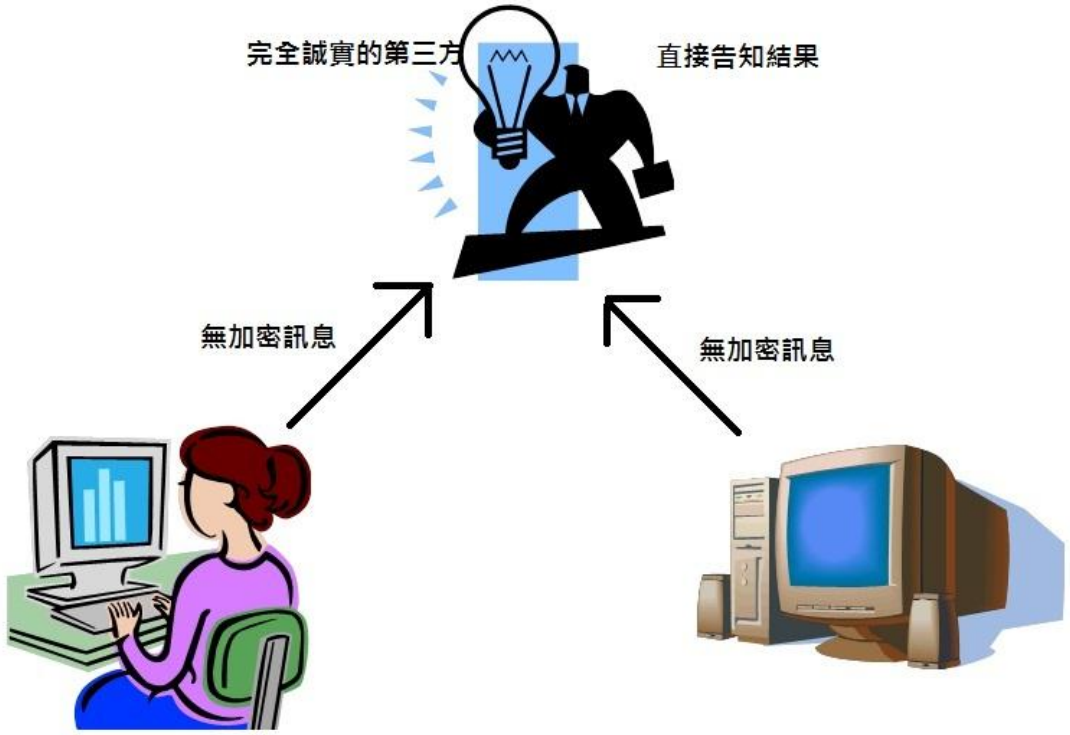

圖 6 誠實的第三方

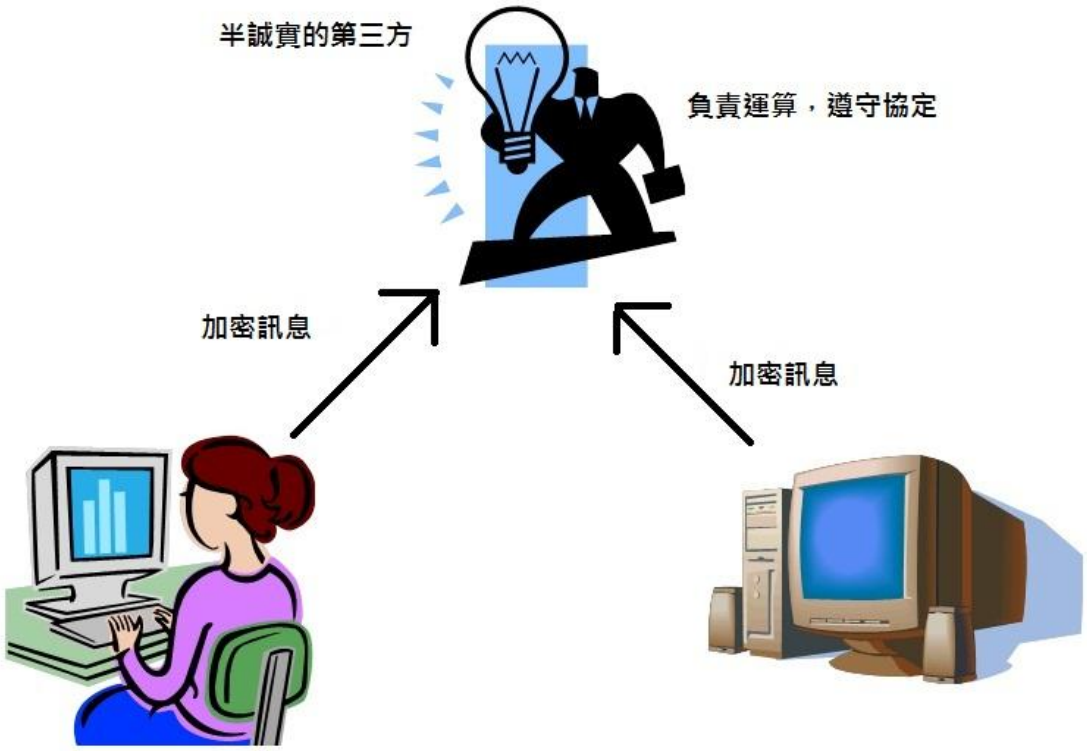

圖 7 半誠實的第三方

2.7.2 Co-operative Private Equality Test

在 **Co-operative Private Equality Test** [14]文獻我們可以看到藉由半誠實的第三方可以比較容易達到兩數的比較。

Li 和 Wu 的 Protocol：

假設參與者 A 和 B，C 為半誠實的第三方。C 生成公開鑰匙和私密鑰匙(*pk*,*sk*)，A 和 B 生成亂數$r_{ab} \in [0, 2^k - 1]$，$k \geq 80$：

1. A 加密自己的秘密訊息 a，$x=E_{pk}(r_{ab}a)$。

把 x 傳給 B。

2. B 加密自己的秘密訊息 b，$y=E_{pk}(-r_{ab}b)$。

計算$y' = E_{pk}(r_{ab}a)E_{pk}(-r_{ab}b)$。

然後 B 把y'傳給 C。

3. C 把y'解開，$z=D_{sk}(y')$。

如果 $z=0$ 傳$z' = 0$給 A 和 B，如果不等於 0 傳$z' = 1$給 A 和 B。

結果：

$z'=0$ 代表 $a=b$，$z' = 1$代表 $a \neq b$

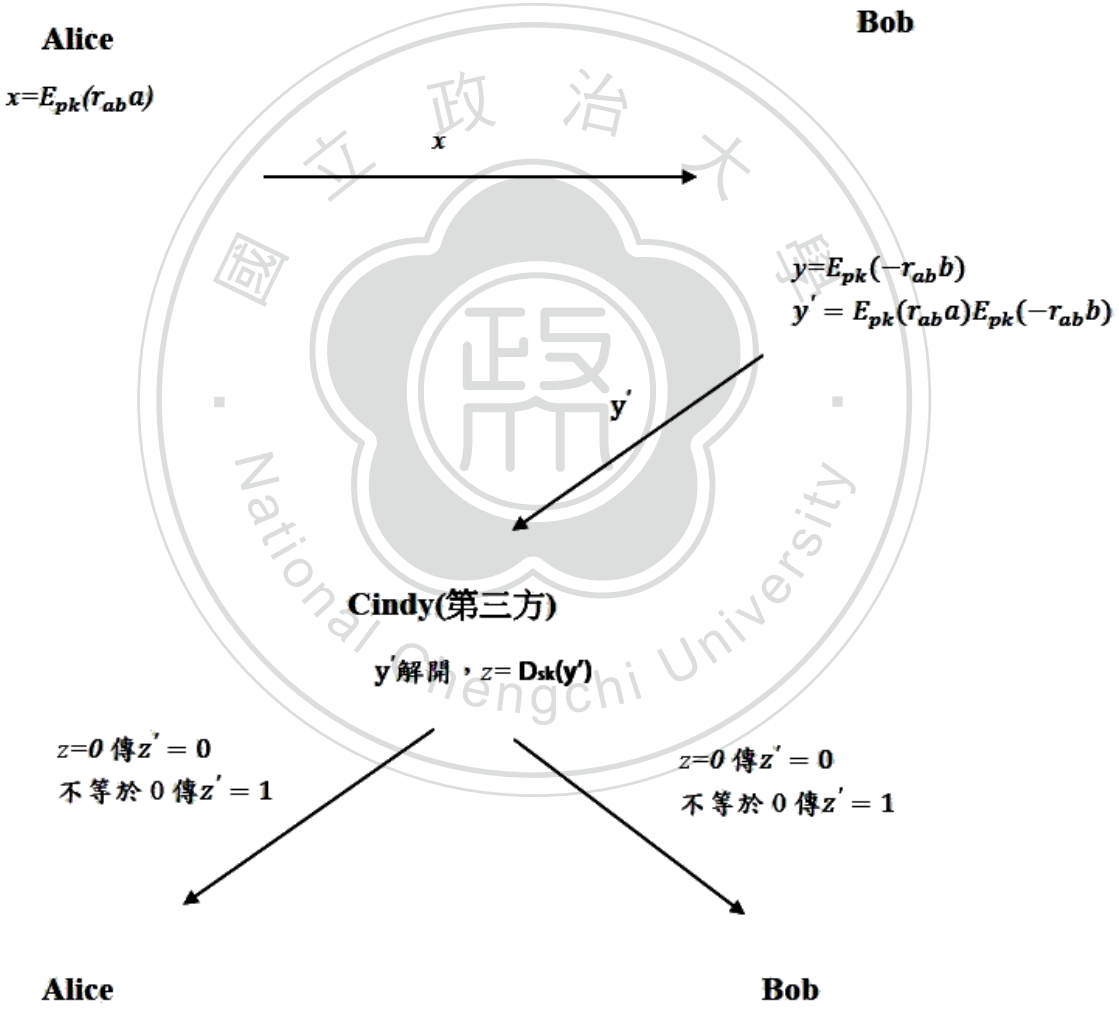

圖 8 第三方協定

問題點：

此方法藉由第三方雖然計算上比較不複雜，而且也步驟也不那麼繁複，但假如此第三方(C)跟任何一方(A or B)合作的話，那麼就容易的拿到另一方的資訊，所以到底要如何找到這可信任的第三方就會是個問題，由於這個問題我們的方法還是研究以雙方溝通的情況，畢竟多一個人就有多一分風險，除非每個人所握的資訊不會因為其中的兩方合謀就能解開，或者能夠確保不會有此問題的第三方協定，當然這也是我們可以探討的，但本篇文章還以雙方為主軸。

第三章 研究方法

本篇研究的主要目的為解決 2.6 和 2.7 章節的問題點,且達成比較兩數值是否相等。本研究為了解決兩個問題點，以 2.7 來看，要再透過一個人才能得到結果，本身就是有一定的風險，所以本研究是以雙方秘密計算為架構來改良，這樣就不用考慮找第三方，也解決了 2.7 的問題。由於是以雙方秘密計算來設計，為了解決 2.6 的問題，本研究設計一個協定，此協定能讓被告知方可以驗證告知方給的結果是否正確，這樣也就解決了 2.6 的問題。

本章節先以雙方秘密計算來設計，所以此章節並沒有驗證方法，在第四章會以此基本架構來改良設計一個有驗證方法的協定。

3.1 基礎協定設計

首先，本研究方法先以原本雙方秘密計算來設計，我們先介紹一個符號：

E 是為一個符合 Doubly-homomorphic encryption 的同態加密演算法並有 Semantic security 的特性。也因為需要達到 Doubly-homomorphic 所以我們可以假設 E 為 Gentry 的加密演算法。另外，我們設定整個協定還需要 $mod\ n$ 來滿足反元素運算和決定訊息長度 N-bit，這個 n 是公開的值且由 Alice 和 Bob 共同決定的質數，N 是公開的值且由 Alice 和 Bob 共同決定的訊息長度，而且 p(Gentry 演算法中的秘鑰)>>2^N(參考下面文章)>>n。使用 Gentry 的加密演算法我們就能直接做乘法的同態運算。

為了使協定中的 r 能有反元素，也就是 $r \in n$。

E_B代表使用 Bob 的秘鑰加密的加密演算法，用此秘鑰加密得到的密文只有 Bob 能解開。

其中加密演算法(參考 2.5 節)中的 $2r$'的 2 改成為2^N，N 為 bit 的值，為了能計算整數，而不是一個 bit。

步驟：

1. Alice 執行

Alice 生成亂數$r = r_x + r_y$ ，$r, r_x, r_y \in Z^*$。

把r_y傳給 Bob。

2. Bob 執行

Bob 收到r_y後，把自己的秘密訊息 b 跟r_y先乘起來得到$r_y b$，再用自己的秘鑰加密起來得到$E_B(r_y b)$。

把$E_B(r_y b)$傳給 Alice。

3. Alice 執行

Alice 收到$E_B(r_y b)$後，把自己秘密的訊息 a 跟r_x相乘，得到$r_x a$。

直接運算$r_x a + E_B(r_y b)$得到$E_B(r_x a + r_y b)$*註[1]。

再來計算 r 在 mod n之下的反元素r^{-1}的值，最後再乘以r^{-1}得到$E_B((r_x a + r_y b)/r)$。

把$E_B((r_x a + r_y b)/r)$傳給 Bob。

4. Bob 執行

Bob 把$E_B((r_x a + r_y b)/r)$解開得到$(r_x a + r_y b)/r$再把值 $mod\ n$ 來達到最後的結果。假如$(r_x a + r_y b)/r = b$，則 Bob 判定$a = b$，假如$(r_x a + r_y b)/r \neq$ b，則 Bob 判定$a \neq b$。

[1]由於使用 Gentry 的加密演算法所以$E_B(r_y b) = r_y b + 2^N r' + pq$，所以

$E_B(r_x a + r_y b) = r_x a + r_y b + 2^N r' + pq = r_x a + E_B(r_y b)$

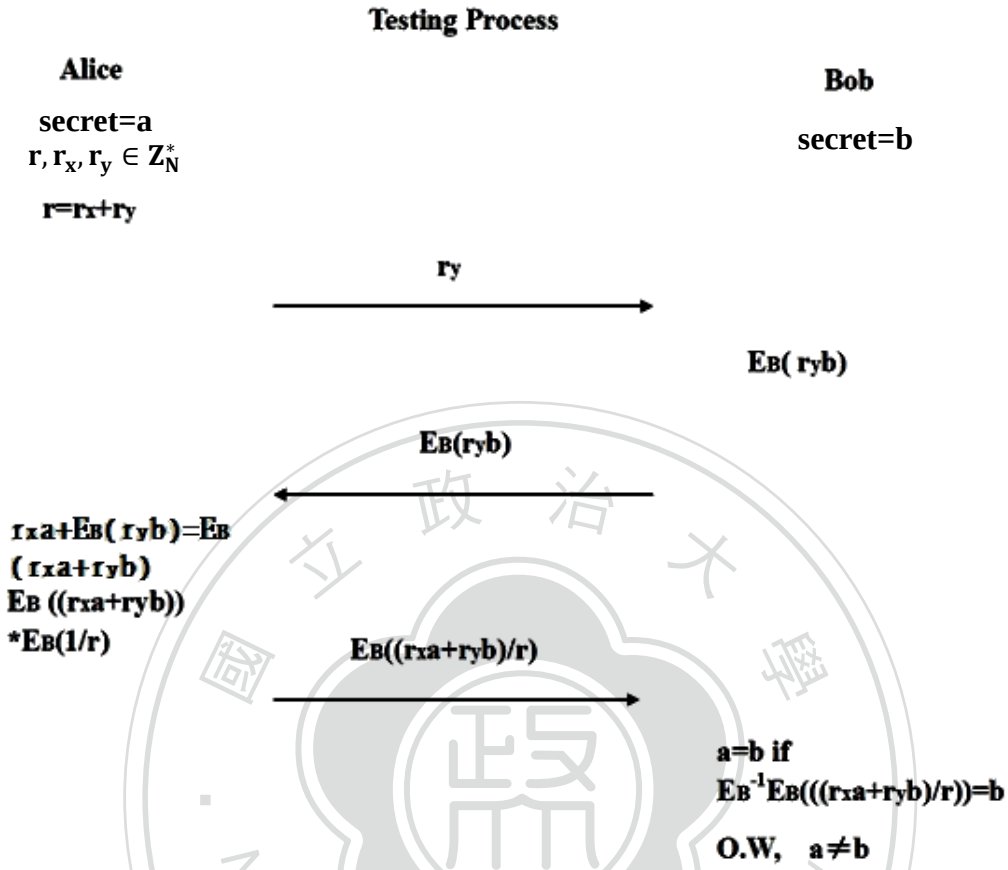

圖 9 雙方秘密計算協定

得知結果之後 Bob 就會把結果告訴 Alice 這也是典型的雙方秘密計算，一方為告知方(Bob)，一方為被告知方(Alice)。

3.2 安全性

在安全性方面，考慮兩種情況，一為 Alice 為攻擊者，另一為 Bob 為攻擊者的情況。

Alice 為攻擊者：

由上面的協議可以知道，只有在第 3 步驟 Alice 才能拿到含有訊息 b 的一個數值 $E_B(r_y b)$，但是假如 Alice 能從$E_B(r_y b)$得到 Bob 的數值 b，這也代表 Alice 能以僅知密文

的攻擊方式破解此密碼系統，但現在的任何安全的公開密碼系統必須能防止這種僅知密文攻擊法，所以代表 Alice 不能從此得到 Bob 的數值 b。

Bob 為攻擊者：

在第 4 步驟 Bob 能拿到含有訊息 a 的一個數值$E_B((r_x a + r_y b)/r)$。

由於 Bob 解開鑰匙能得到$(r_x a + r_y b)/r$的數值，所以我們設為 t，所有數直接在 *mod n* 環境下進行，亦即：

$$(r_x a + r_y b)/r = t$$

又$r = r_x + r_y$，所以：

$$(r_x a + r_y b)/(r_x + r_y) = t$$

移項後得：

$$(r_x a + r_y b) = t(r_x + r_y)$$

從第 2 步驟可知，Bob 知道r_y的數值，我們再加以整理：

$$(r_x a - r_x t) = (r_y t - r_y b)$$

$S = (r_y t - r_y b)$，S是 Bob 已知

$$(r_x a - r_x t) = S$$

最後得到：

$$r_x(a - t) = S$$

$$r_x(Z) = S \text{，} Z = a - t$$

我們可看到r_x和 Z 對於 Bob 都是未知的，所以當 Bob 猜測一個r_x的值，也會對應一個 Z 的值，滿足$r_x(Z) = S$，所以 Bob 並不能從中得到 Alice 的數值 a。

3.3 應用方面

雖然此方案沒有讓 Alice 驗證的機制，但實務上還是有其應用價值的。在這種情況之下，我們可以是使用在身分認證，也就是一個使用者登入這種需要比較 ID 和 password 是否相等的情況，在這種情況下 A 方就不需要再對最後的結果確認，而我們的基本方案就可以使用在這種情況之下。

下圖為傳統使用者端登入情況。

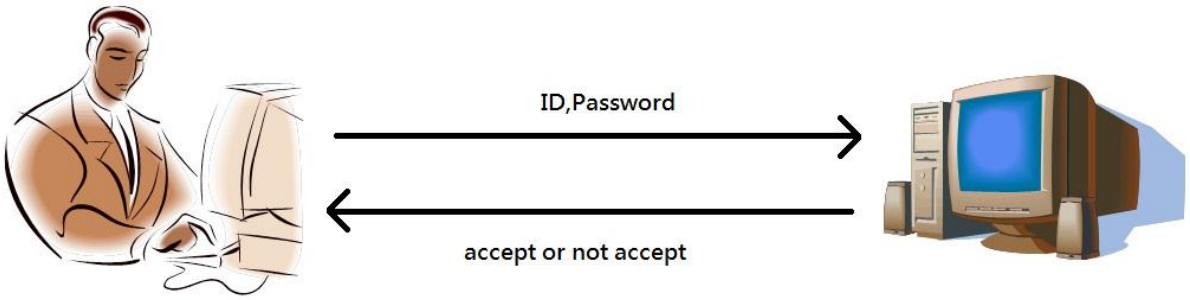

圖 10 傳統登入系統

使用者輸入 ID 和 Password 後，系統就使用這兩個訊息到資料庫中搜尋通行密碼檔案進行比對，正確允許使用者登入系統。

缺點：

1. 傳輸過程洩露相關訊息。

在傳輸的過程，傳輸 password 並不是以加密的狀態下傳輸，所以當竊取者竊取資訊，可以直接拿到秘密的訊息。

2. 使用單向雜湊函數(One Way Hash Function)。

舉例說明，假如我們的 password 是 111，經過 hash function 加密之後是 123456，由此可見，111 經過 hash function 一定是 123456，所以竊取者只要能找到符合或竊取到 123456 這個數值，竊取者就能登入，在安全性還是有點危險。

以本研究協定架構下，所產生的登入系統，如下圖。

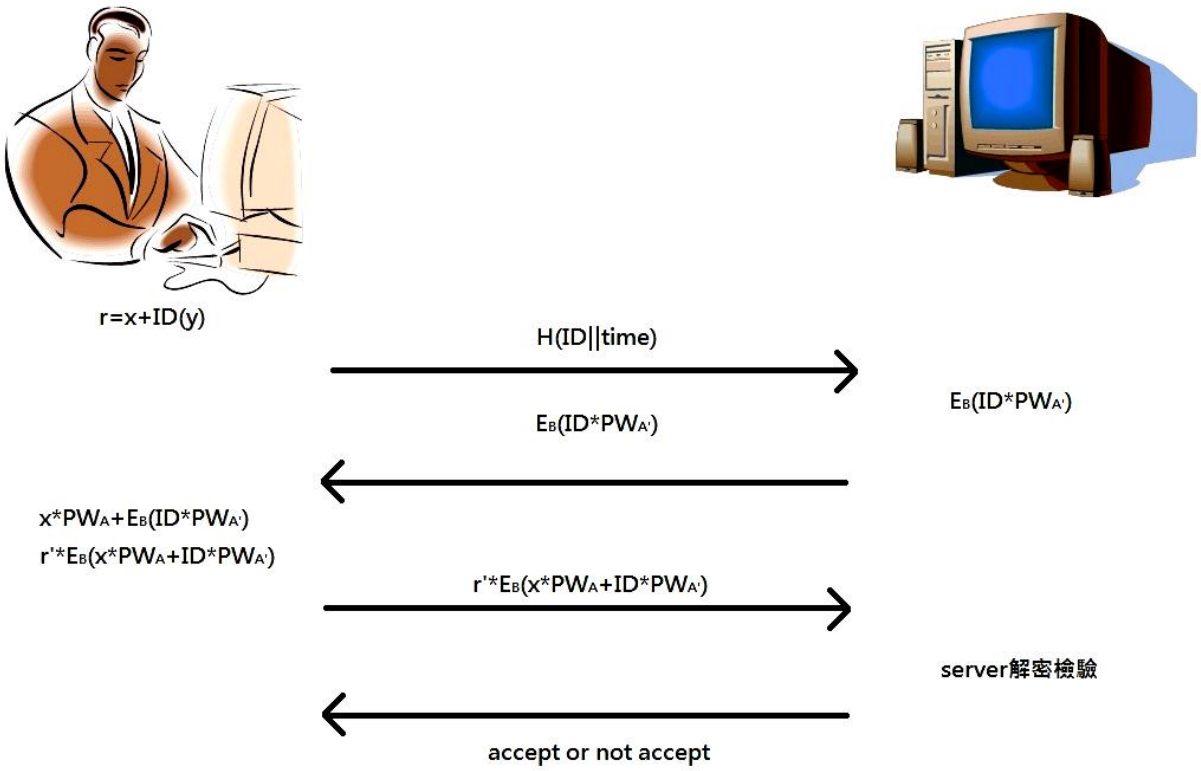

圖 11 本研究登入系統架構

解決問題：

1. 本登入系統的傳輸過程中，所有的秘密資訊都是經過加密，所以竊取者想從傳輸過程中竊取資訊是不可行的。

2. 本登入系統使用的加密方法為同態加密，相同數值經過加密後的數值都不一樣。比起傳統的登入系統，本系統的安全性是比較好的，竊取者一定要知道 password 才能登入。

第四章　雙方相等性驗證機制

在之前的章節有介紹過，在傳統的雙方協定中，被告知方只能從告知方得知結果，就像 3.1 章節所設計的基本協定。在不同的情況或應用之下，被告知方也想確認告知方到底告知的結果是否屬實，所以本章節就以 3.1 的協定再加以改進，來達到讓被告知方可以驗證告知方告知之結果，讓整個架構更加完整，也解決了 2.6 和 2.7 章節的問題。由於有讓被告知方可以驗證告知方告知之結果的功能，所以也讓參與的雙方更有公平性。

4.1 協定設定

E 是一個符合 Doubly-homomorphic encryption 的同態加密演算法並有 Semantic security 的特性。除此之外，還需要符合 Commutative encryption。為了同時達到 Doubly-homomorphic encryption 和 Commutative encryption，所以要利用 Gentry 的加密演算法，並做些變化，在之後的文章會說明。

E_A代表使用 Alice 的公鑰的同態加密演算法，用此公鑰加密得到的密文只有 Alice 能解開。。E_B代表使用 Bob 的公鑰的同態加密演算法，用此公鑰加密得到的密文只有 Bob 能解開。

步驟：

1. Alice 執行
   - 生成亂數$r = r_x + r_y$ ，$r, r_x, r_y \in Z^*$。
   - 把自己的秘密訊息 a 和r_x相乘再用自己的公鑰加密起來得到$E_A(r_x a)$。
   - 把r_y和$E_A(r_x a)$傳給 Bob。

2. Bob 執行
   - 收到r_y和$E_A(r_x a)$後，把自己的秘密訊息 b 跟r_y先乘起來得到$r_y b$，再用 Alice 的公鑰加密起來得到$E_A(r_y b)$。
   - 再使用前面 2.2 章節所介紹的同態性質把$E_A(r_x a)$和$E_A(r_y b)$做加法運算得到$E_A(r_x a + r_y b)$。
   - 亂數選擇$q \in Z^*$，用乘法同態把$E_A(r_x a + r_y b)$乘$E_A(q)$，這樣就會得到$E_A(q(r_x a + r_y b))$。
   - 用自己的同態公鑰加密起來得到$E_B E_A(q(r_x a + r_y b))$。

- 把$E_B E_A(q(r_x a + r_y b))$傳給 Alice。

3. Alice 執行

- 收到$E_B E_A(q(r_x a + r_y b))$後，先用自己的私鑰解密得到$E_B(q(r_x a + r_y b))$。
- 先計算$E_B(r^{-1})$，再把$E_B(q(r_x a + r_y b))$用乘法同態，乘以$E_B(r^{-1})$，得到$E_B((q(r_x a + r_y b))/r)$。
- 把$E_B((q(r_x a + r_y b))/r)$傳給 Bob。

4. Bob 執行

- 收到$E_B((q(r_x a + r_y b))/r)$後，用自己的私鑰解開得到$(q(r_x a + r_y b))/r$。
- 再把 q 除掉，假如$(r_x a + r_y b)/r = b$，則 Bob 判定$a = b$，假如$(r_x a + r_y b)/r \neq$ b，則 Bob 判定$a \neq b$。

上述的協定，是我們最初的架構，由於需要符合 Doubly-homomorphic encryption，所以我們使用 Gentry 的演算法。也需要符合 Commutative encryption，但因為使用 Gentry 的演算法(並沒有符合 Commutative encryption)，所以我們需要把我們的協定修改一下，來符合 Commutative encryption，又能確保結果的正確性。修改如下：

在 Gentry 的加密法為：$c = m + 2r' + pq$ 可參考 2.5 章節。

Alice 所生成秘鑰為p_a。

Bob 所生成秘鑰為p_b。

其中加密演算法中的 *2r'*的 *2* 改成為2^N，N 為 bit 的數值，這樣就能計算整數，而不是一個 bit。

最後規定$p_a \gg p_b \gg 2^N \gg p'$，$p' \in Z^*$為一個質數，由 Alice 和 Bob 共同決定，為了使協定中的 r 能有反元素，也就是 $r \in p'$。

E_A代表使用 Alice 的秘鑰加密的加密演算法，用此秘鑰加密得到的密文只有 Alice 能解開。。E_B代表使用 Bob 的秘鑰加密的加密演算法，用此秘鑰加密得到的密文只有 Bob 能解開。

協定步驟：

1. Alice 執行
   - 生成亂數$r = r_x + r_y$ ，$r, r_x, r_y \in Z^*$。
   - 把自己的秘密訊息 a 和r_x相乘，再用自己的秘鑰加密起來得到$E_A(r_x a)$。
   - 把r_y也用自己的秘鑰加密，得到$E_A(r_y)$。
   - 把$E_A(r_y)$和$E_A(r_x a)$傳給 Bob。

2. Bob 執行
   - 收到$E_A(r_y)$和$E_A(r_x a)$後，把自己的秘密訊息 b 跟$E_A(r_y)$先乘起來得到$E_A(r_y b)$，使用 Gentry 的方法可以直接乘不會影響解密後的值。
   - 而我們知道 Gentry 符合 Doubly-homomorphic encryption，把$E_A(r_x a)$和$E_A(r_y b)$做加法運算得到$E_A(r_x a + r_y b)$。
   - 亂數選擇$q \in Z^*$，把$E_A(r_x a + r_y b)$乘以 q，這樣就會得到$E_A(q(r_x a + r_y b))$。
   - 再用自己的秘鑰加密起來得到$E_B E_A(q(r_x a + r_y b))$。
   - 把$E_B E_A(q(r_x a + r_y b))$傳給 Alice。

3. Alice 執行
   - 收到$E_B E_A(q(r_x a + r_y b))$後，先用自己的秘鑰解密得到$E_B(q(r_x a + r_y b))$
   - 再來計算 r 在 $mod\ p'$之下的反元素r^{-1}的值，再把$E_B(q(r_x a + r_y b))$乘以 r^{-1}，得到$E_B((q(r_x a + r_y b))/r)$。

- 把$E_B((q(r_x a + r_y b))/r)$傳給 Bob。

4. Bob 執行

- 收到$E_B((q(r_x a + r_y b))/r)$後，用自己的秘鑰解開然後需要再 $mod\ p'$才能得到$(q(r_x a + r_y b))/r$。
- 再把q除掉，假如$(r_x a + r_y b)/r = b$，則Bob判定$a = b$，假如$(r_x a + r_y b)/r \neq$ b，則 Bob 判定$a \neq b$。

4.2 驗證方法

在之前的章節就有闡述過，我們其中之一的目的在於讓被告知方也能驗證告知方給的結果是否正確。

驗證如下：

1.

在 Bob 計算最後結果之後，需要把第 4 步驟的$(q(r_x a+r_y b)/r) \bmod p'$和第 2 步驟的$E_A(q(r_x a+r_y b))$傳給 Alice。

2.

Alice 收到$(q(r_x a+r_y b)/r) \bmod p'$和$E_A(q(r_x a+r_y b))$後，計算 M=解開$E_A(q(r_x a+r_y b)= (q(r_x a+r_y b))$，Alice 就能判斷 M 是否等於第 4 步驟的$(q(r_x a+r_y b)/r) \bmod p'$乘以 r。

3.

更進一步的判斷，假如$a = b$代表，$a|M/r$；否則$a \neq b$。

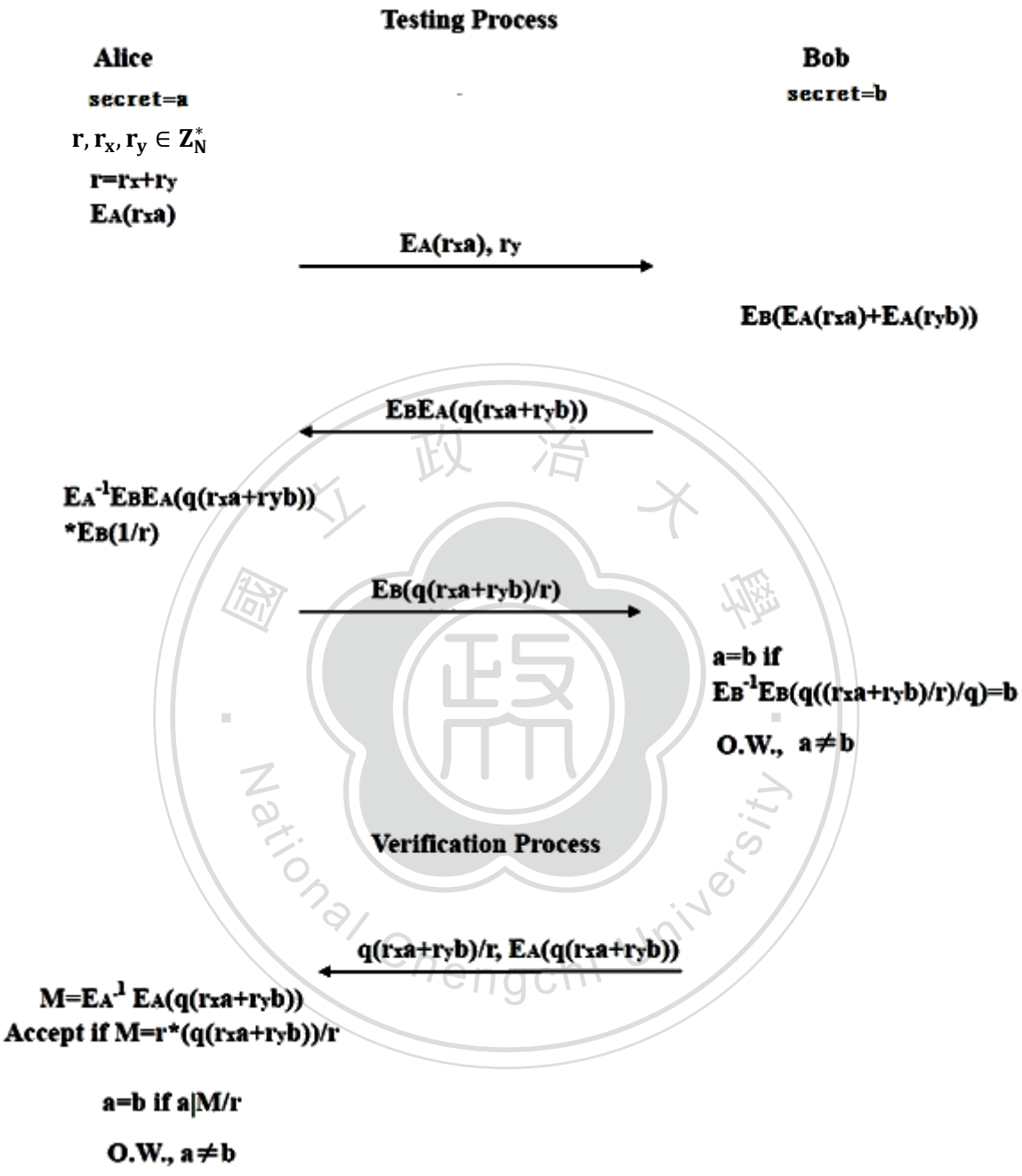
Testing Process
Alice
secret=a
r, r_x, r_y ∈ Z_N^*
r=r_x+r_y
$E_A(r_xa)$
$E_A(r_xa)$, r_y
Bob
secret=b
$E_B(E_A(r_xa)+E_A(r_yb))$
$E_BE_A(q(r_xa+r_yb))$
$E_A^{-1}E_BE_A(q(r_xa+r_yb))$
*$E_B(1/r)$
$E_B(q(r_xa+r_yb)/r)$
a=b if
$E_B^{-1}E_B(q((r_xa+r_yb)/r)/q)=b$
O.W., a≠b
Verification Process
$q(r_xa+r_yb)/r$, $E_A(q(r_xa+r_yb))$
$M=E_A^{-1}$ $E_A(q(r_xa+r_yb))$
Accept if M=r*(q(r_xa+r_yb))/r
a=b if a|M/r
O.W., a≠b

圖 12　協定與驗證流程

4.3 安全分析

我們一樣考慮 A 為攻擊者和 B 為攻擊者這兩種情況，再對正確性、隱私性和公平性做分析。

A 方為攻擊者：

由於整個協定，只有第二步驟 Alice 才能拿到$E_B E_A(q(r_x a + r_y b))$這個數值含有 Bob 的訊息 b。

我們假設在第三步驟，Bob 給定：

b_m，$m \in \{0,1\}$，所以傳給 Alice$E_B E_A(q(r_x a + r_y b_0))$和$E_B E_A(q(r_x a + r_y b_1))$這兩個數值，而我們一開始就有說明此加密演算法滿足 Semantic security 且 Bob 還多加了 q 這一亂數，所以我們能夠確保 Alice 是無法分辨$E_B E_A(q(r_x a + r_y b_0))$和$E_B E_A(q(r_x a + r_y b_1))$到底各為是$b_0$還是$b_1$所產生的數值，由此可知 Alice 無法從這一步的到任何關於 b 的訊息。

另外要探討的是在於驗證步驟 Alice 能拿到$(q(r_x a+r_y b)/r)$和$E_A(q(r_x a+r_y b))$，後面的數值解密過後其實跟前面數值只相差 r 倍，所以我們拿 $q(r_x a+r_y b)/r$ 來做分析：

Alice 已知r_x、a、r_y和 r，假設：

$$q(r_x a+r_y b)/r=M$$

我們可以發現此式子有兩個未知數q和b，我們知道一個式子並沒有辦法解兩個未知數，所以此情況 Alice 並沒有辦法解開得到 Bob 的數值 b。

B 方為攻擊者：

在第二步驟 Bob 能拿到$E_A(r_x a)$這含有 a 值的數值。

假設$E_A(r_x a_m) = L$，Alice 給定：

a_m，$m \in \{0,1\}$，由於此加密演算法滿足 Semantic security，這代表 Bob 並不能從L來判斷 Alice 是由a_0還是a_1加密而來的，所以無法得知 Alice 的數值 a。

在第六步驟 Bob 能拿到$E_B((q(r_x a + r_y b))/r)$這含有 a 的數值。

我們解密得到$q(r_x a + r_y b))/r$，以此證明，如下：

以下數值都是在 $mod\ p'$的環境下進行，我們假設：

$$q(r_x a + r_y b))/r = M \quad M\text{為已知數值}$$

q 是由 Bob 產生的，所以：

$$(r_x a + r_y b))/r = M/q = S \quad S\text{為 Bob 已知數值}$$

又$r = r_x + r_y$：

$$(r_x a + r_y b)/(r_x + r_y) = S$$

移項後得：

$$(r_x a + r_y b) = S(r_x + r_y)$$

從第 2 步驟可知，Bob 知道r_y的數值，我們再加以整理：

$$(r_x a - r_x S) = (r_y S - r_y b)$$

$$P = (r_y S - r_y b) \text{，} P\text{是 Bob 已知}$$

$$(r_x a - r_x S) = P$$

最後得到：

$$r_x(a - S) = P$$

$$r_x(Z) = P \text{，} Z = a - S$$

我們可看到r_x和 Z 對於 Bob 都是未知的，所以當 Bob 猜測一個r_x的值，也會對應一個 Z 的值，滿足$r_x(Z) = S$，所以 Bob 並不能確定r_x，藉而得到 Alice 的數值 a。

正確性：

我們可以看到，無論是 Alice 或者是 Bob 只要照著協定的規則進行，Bob 最後會得到*(a+b)/r* 只要 *a* 和 *b* 相等 Bob 就會得到 *b* 這個值，也就是代表 *a* 和 *b* 相等，而 Alice 只要 Bob 依照協定把 *q(a+ b)/r* 的值和 *q(a+b)/r)*，這樣 Alice 也能得到正確的值。

隱私性：

藉由上面的證明，我們可以確保 Alice 和 Bob 都沒有洩漏秘密，所以我們能確定 Alice 和 Bob 都能達到隱私性。

公平性：

在 Bob 得到 *a* 不等於 *b* 的情況之下，Alice 一樣也可以的到此結果，由於我們協定提供驗證的方法，所以 Bob 並不能以欺騙的結果告訴 Alice，同理，Bob 得到 *a* 等於 *b*，Alice 也可以得到此結果。

第五章　效能分析

在這一章節，對我們所提出的協定寫一個簡單的程式，而此程式為測試在此協定下所需的計算時間，我們使用的演算法是 Gentry 的演算法，然後如同上一章節提到改良方法來實行。

程式內容只有包含各步驟的計算時間，用此說明我們的協定並不會耗費太多時間在計算方面。傳輸方面，以現在發達的網路系統，傳送一、兩的數值也不會運用到太多時間，所以本程式也沒特別的去計算此方面。

在協定中有些亂數的產生，跟一些先自己決定或者共同決定的值，如：訊息 a、亂數$r = r_x + r_y$、p_a(Gentry 加密所需生成的)……等，我們也不列入計算時間內，所以我們就如此分析計算，到底需要多少時間計算來完成整個協定。

此章節是以第四章所介紹的協定來做分析，由於第四章的協定是以第三章的協定加以改良，以整個協定來看，第四章的計算量是大於第三章，所以此章節是以分析第四章協定來敘述。

5.1 程式架構

整個協定我們分成四個部分，

第一(A1)：

計算$E_A(r_x a)$和$E_A(r_y)$

第二(B1)：

計算$E_A(r_y b)$、$E_A(r_x a + r_y b)$、$E_B E_A(q(r_x a + r_y b))$

第三(A2)：

計算$E_B(q(r_x a + r_y b))$和$E_B((q(r_x a + r_y b))/r)$

第四(B2)：

計算$(q(r_x a + r_y b))/r$ 和 *mod* p'之後的值

而我們以p_a、p_b、*N*(是 Gentry 裡的$2^N r'$的 *N*)來看計算時間

1. 在p_a長度一樣為 500-bit，N 長度一樣為 20-bit，p_b各為 60-bit 和 250-bit 的情況，如下圖

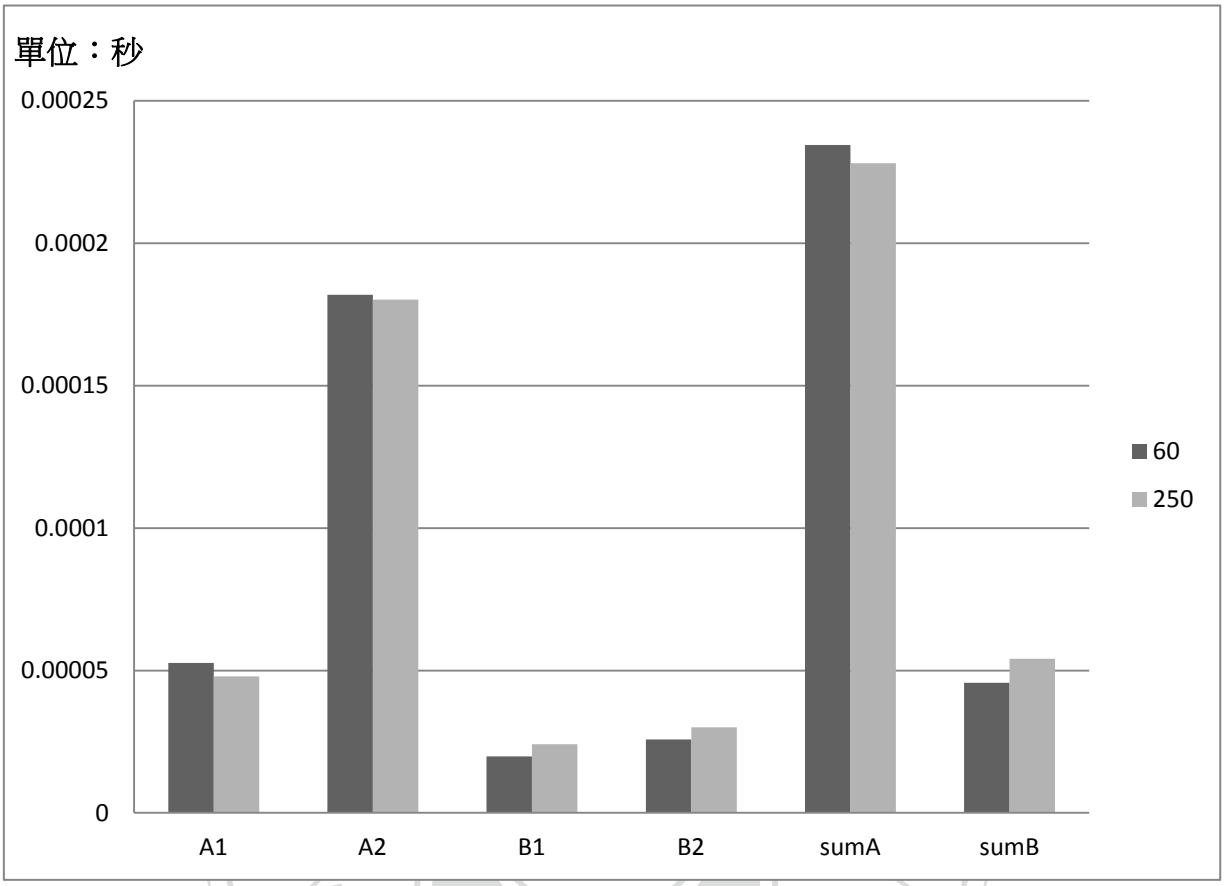

圖 13 數據 1

時間上的單位是奈秒(10^{-9})，由圖 13 可知，基本上運算時間都沒太大落差，由於$p_b$bit 數增加所以在 Bob 所要計算時間比較多一些，但基本上是沒有甚麼差距的，整體計算時間 $A>B$，可以看出整個計算時間是蠻小的。

2. 在p_a長度各為 100-bit、500-bit，N 長度一樣為 20-bit，p_b為 60-bit 的情況，如下圖

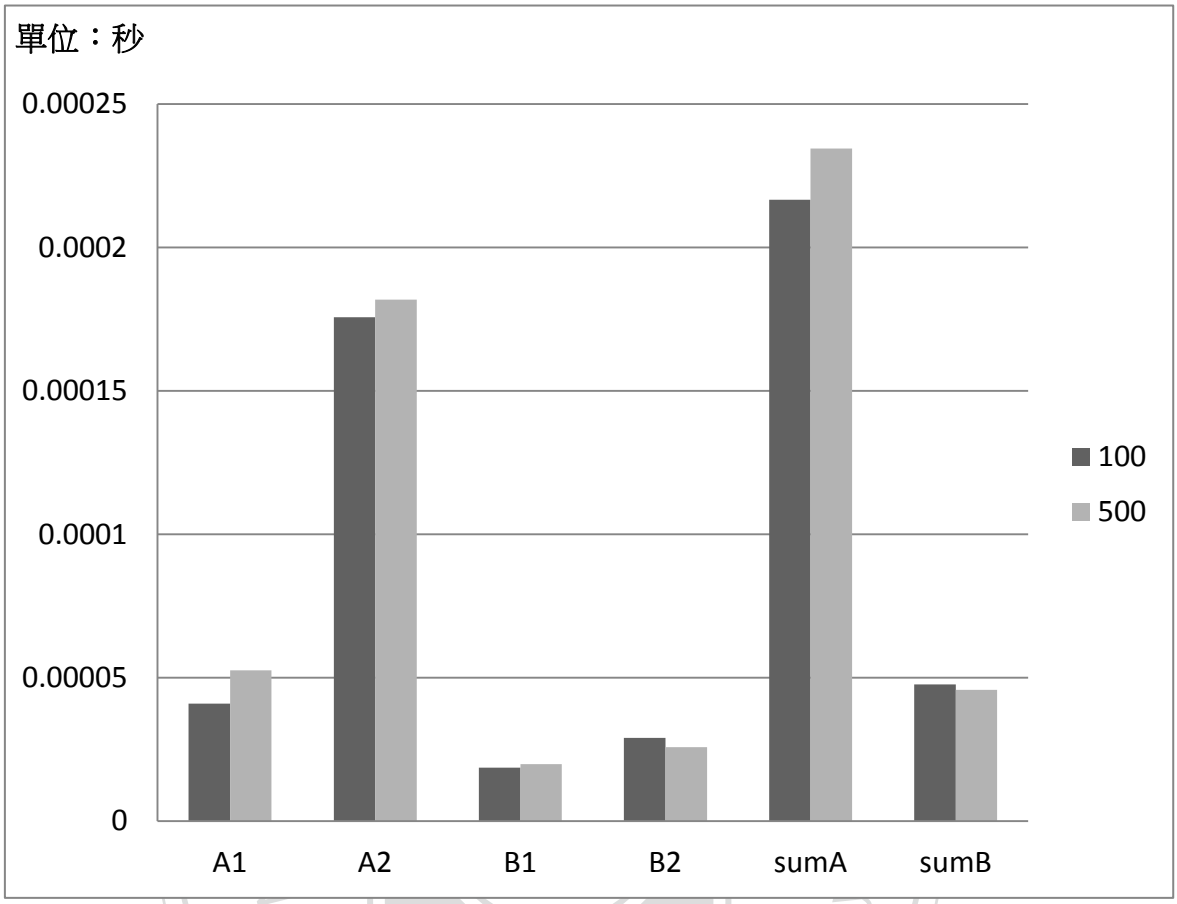

圖 14 數據 2

圖形跟上張圖蠻類似的，幾乎差距不大，可以看出跟p_a比較有關的(A1 和 A2)有明顯差距。

3. 在p_a長度為 500-bit，N 長度各為為 20-bit、50-bit、100-bit，p_b為 250-bit 的情況，如下圖

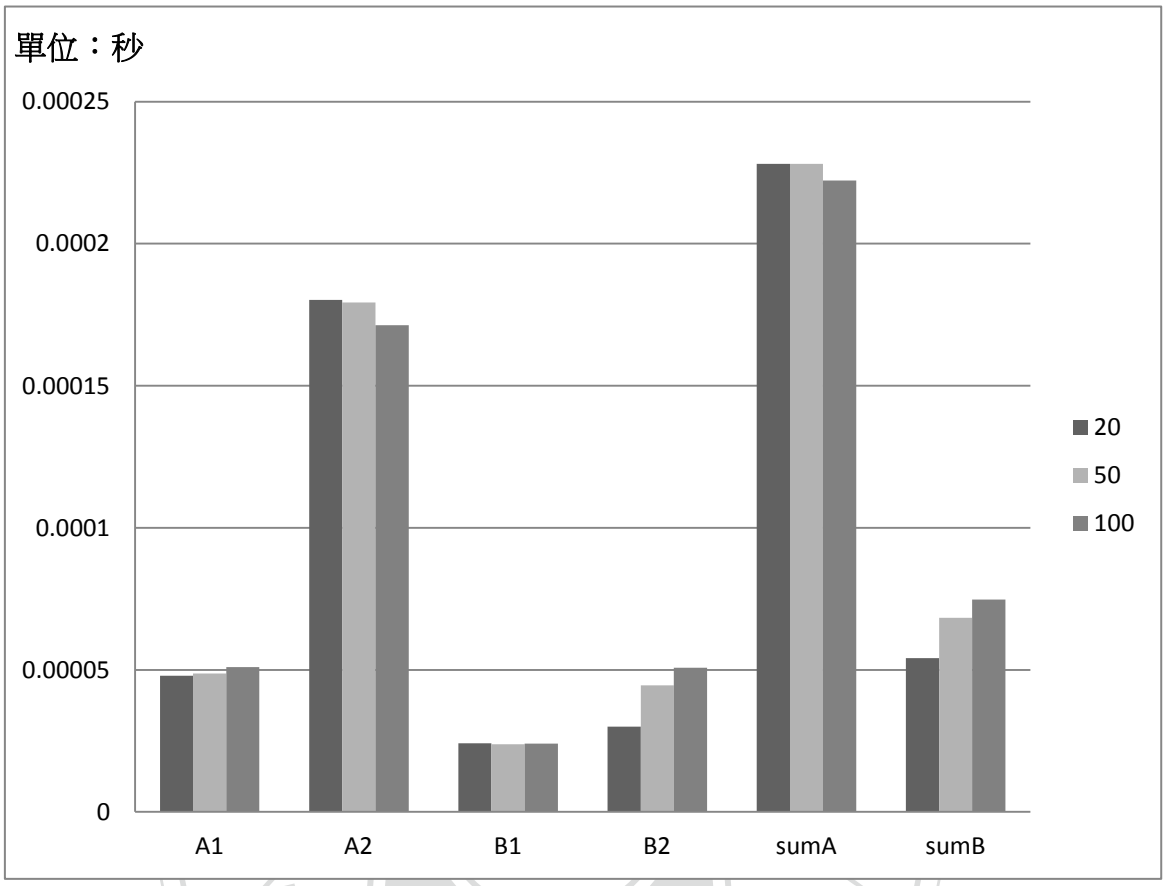

圖 15 數據 3

可以看出 A2 跟 N 的計算沒有關系所 N 的改變跟 A2 的變化是沒有關係的，而 B2 會用到 N 所以有增加的情況。

整體來講，整個協定的計算時間非常的小，A2 的時間跟 B2 的時間是比較多的，代表有 *mod* 的計算量比較大，也代表解密的情況比較費時。

第六章 結論

本論文中提出了一個計算雙方秘密數值是否相等的機制，並證明其協定的安全性，而在效能上，雖然沒有相關研究的比較，但可以從數據知道計算時間花費很小。本研究提出以雙方來達成比較秘密數值是否相等，這樣就不需要依靠半誠實的第三方來達成數值比較。本研究有驗證的方法，使得傳統雙方協定常有被告知方知能從告知方得知結果的問題得以解決，這樣也讓被告知方對於最後的結果可以更加的信任。

未來我們也希望可以延續此研究想法，以更簡易了解的同態加密演算法，來達到我們的需求，如：RSA、ElGamal……等。也希望以後有更容易的方法，來達到雙方數值的比較，使現在的網路時代上，任何有關數值比較，如：登入、交易……等相關系統，都能運用我們的方法，更加安全的使用。

參考文獻

[1]. D. Boneh, E. Goh, and K. Nissim. Evaluating 2-dnf formulas on ciphertexts. In Proceedings of Theory of Cryptography (TCC),2005, pp. 325-341.

[2]. A. Beimel, T. Malkin, S. Micali. The all-or-nothing nature of two-party secure computation. In Proceedings of CRYPTO 99, 1999, pp. 80-97.

[3]. E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems, Journal of Cryptology, Vol.4, No.1, 1991, pp. 3-72.

[4]. N. Courtois and J. Pieprzyk: Cryptanalysis of Block Ciphers with Overdefined Systems of Equations, Asiacrypt ,2002, LNCS 2501, pp.267-287.

[5]. B. Chevallier-Mames, J. Sebastien Coron, N. McCullagh, D. Naccache, and M.Scott. Secure delegation of elliptic-curve pairing. Cryptology ePrint Archive, 2005, pp.24-35.

[6]. T. Chiang,W. Wang, J. Liau, and -S. Hsu. Secrecy of two-party secure computation. Lecture Notes in Computer Science, 2005, pp. 114-123.

[7]. W. Diffie and M. Hellman, "New directions in cryptography," IEEE Trans. Inform. Theory, vol. IT-22, 1976, pp. 472-492.

[8]. W. Du and Z. Zhan. A practical approach to solve secure multiparty computation problems. In Proceedings of New Security Paradigms Workshop, 2002, pp. 127-135.

[9]. T. E. Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In proceedings of CRYPTO, 1985, pp. 10-18.

[10].R. Fagin, M. Naor, P. Einkler, Comparing information without leaking it, Communications of the ACM 5 , 1996, pp.77-85.

[11]. C. Gentry. Fully homomorphic encryption using ideal lattices. In STOC ’09, ACM, 2009, pp. 169–178.

[12].O. Goldreich, S. Micali and A. Wigderson, How to play any mental game or a completeness theorem for protocols with honest majority. In Proceedings of the 19th Annual ACM Symposium on Theory of Computing (STOC), 1987, pp. 218-229.

[13]. S. Goldwasser and S. Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In Proceedings of the 14th ACM Symposium on Theory of Computing (STOC’82), 1982, pp. 365–377.

[14].R. Li and C. K.Wu, Co-operative private equality test. International Journal of Network Security, vol.1, no.3,2005, pp. 149-153.

[15].D. Naccache and J. Stern. A new public key cryptosystem based on higher residues. In Proceddings of Computer and Communications Security (CCS), ACM, 1998, pp. 59-66.

[16].C. P. Schnorr. E_cient Identi_cation and Signatures for Smart Cards. In Crypto '89, LNCS 435, 1990, pp. 235-251.

[17].P. Paillier. Public-key cryptosystem based on composite degree residuosity classes. In Proceedings of Eurocrypt 99, 1999, pp. 223-238.

[18].R. Rivest, A. Shamir, L. Adleman . A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM **21** (2), 1978, pp. 120–126.

[19].J. Vaidya and C. Clifton. Leveraging the ”Multi” in Secure Multi-Party Computation. In Proceedings of the Workshop on Privacy in the Electronic Society, 2003, pp. 53-59.

[20].B. Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd ed. (Wiley, 1996).

[21].C. Yao, Protocols for secure computation. In Proceedings of the 23rd IEEE Symposium on Foundations of Computer Science (FOCS), 1982, pp. 160-164.

[22].F. Zhang, R. Safavi-Naini, and W. Susilo. An efficient signature scheme from bilinear pairings and its applications. In Proceedings of Public Key Cryptography (PKC) ,2004,pp. 277-290.