銀行業務上的加密運用

台北富邦銀行　徐銘宏副總

2022/1/20

Agenda

- 兩個故事：竊符救趙 ； 日昇昌銀號
- 基礎密碼學：對稱式與非對稱式加密技術
- 銀行的加密運用 ：TDES / RSA

兩個故事：竊符救趙

竊符救趙：

邯鄲之戰發生於公元前259年 - 前257年年間長平之戰的延續。

趙公子平原君多次向魏安釐王和魏無忌送信救援，魏安釐王派將軍晉鄙領兵十萬前去救趙。秦昭襄王派使者威脅魏王「敢救趙者，拔趙後必擊之」，魏安釐王懼怕，就派人通知晉鄙停止進軍。

信陵君其後聽信隱士侯嬴的計謀，先請魏王寵妃如姬從魏安釐王的臥室內竊出虎符，再帶同勇士朱亥至魏軍見晉鄙，拿出兵符假傳魏王的命令要代替晉鄙領軍。晉鄙合了兵符，驗證無誤，但還是表示懷疑，不想交出兵權。信陵君唯有讓朱亥用鐵椎殺死晉鄙，強行奪權。

結果，秦國受到了自商鞅變法後少有之一場大敗，使六國得以延續三十年國祚。

虎符是源於中國的金屬製的虎形調兵憑證。

傳說是西周姜子牙所發明，在中國古代皇帝制度中是軍權的象徵。

兩個故事：日昇昌銀號

日昇昌，清朝著名金融機構，有清朝第一大票號之稱。

日昇昌分號分佈全國不同城市進行錢鈔匯兌的業務，必須製作具有嚴格保密性的匯票。

除了採用當時 世界上最先進的印刷技術及在關鍵部位加蓋戳印外，並實行一套以漢字代表數字的密碼法。

「中國票號博物館」裏，展示了一份「防假密押」，表面上是56個字的順口溜，實則是暗藏玄機。是兌換匯票的一套密碼。

1至12個月：「謹防假票冒取， 勿忘細視書章」 。

1至30天 ：「堪笑世情薄，天道最公平。昧心圖自私，陰謀害他人。善惡終有報，到頭必分明」 。

數字1至10 ：「生客多察看，斟酌而後行」 。

位數 ： 萬千百十「國寶流通」。

例如票號在6月10日給某省票號分號匯銀3000兩，其暗號代碼就是「取平多寶」

- **人：掌櫃操守 & 筆跡**
- **事：押密 & 變更對照表 & 匯票銷毀**
- **物：印刷技術 特殊用紙 & 印戳 & 匯票序號**

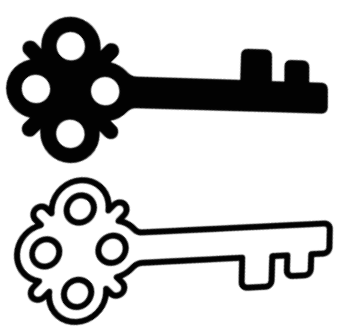
基礎密碼學(對稱式與非對稱式加密技術)

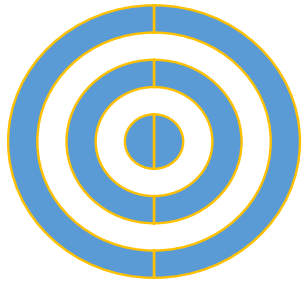
現代密碼學泛指：透過數學演算法與電腦科學對資料進行加密和解密的科學。

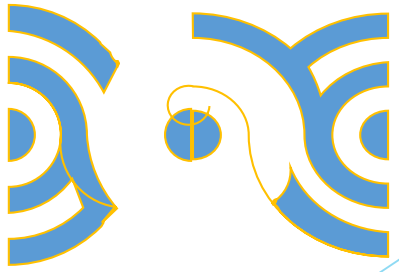
密碼學(Cryptography)必須包含：

- 機密性(Confidentiality)：確保訊息只有被授權者才能取得
- 完整性(Integrity)：偵測訊息是否遭受竄改
- 身分認證(Authentication)：傳送方與接受方需驗證識別
- 不可否認性(Non-Reputation)：提供訊息傳送方與接受方的交易證明

在基礎密碼學中，有兩種加解密方式，分別式「對稱式加密(Symmetric Encryption)」與「非對稱式加密(Asymmetric Encryption)」。







對稱密鑰演算法（Symmetric-key algorithm）

這類演算法在加密和解密時使用相同的密鑰。事實上，這組密鑰在兩個或多個成員間的共同持有。常見的對稱加密演算法有**AES**、ChaCha20、**3DES**、Salsa20、**DES**、Blowfish、IDEA、RC5、RC6、Camellia。

DES: 資料加密標準（Data Encryption Standard）

是一種對稱密鑰 & 密塊密碼演算法。密碼學家霍斯特·費斯妥（Horst Feistel），IBM工作期間完成了此項研究。通常稱為Feistel network/ Feistel cipher。它基於使用56位金鑰的區塊加密對稱演算法。

1976年被美國國家標準暨技術研究院(National Institute of Standards and Technology，NIST)確定為聯邦資料處理標準（FIPS），隨後在國際上廣泛流傳開來。 DES現在已經不是一種安全的加密方法，主要因為它使用的56位金鑰過短。

1999年1月，distributed.net與電子前哨基金會合作，在22小時15分鐘內即公開破解了一個DES金鑰。為了所需的安全性，可以使用DES的衍生演算法TDES來進行加密。

對稱密鑰演算法（Symmetric-key algorithm）

AES進階加密標準(Advanced Encryption Standard)

NIST 1997年向密碼學界徵求能夠替代DES的加密演算法，經過3年的驗證以後，Rijndael演算法最後入選成為進階加密標準。 Rijndael演算法由比利時學者Joan Daemen和Vincent Rijmen提出。

奠定現代密碼學的兩大基柱分別是:
取代-重排網路與費斯妥加密

Substitution-Permutation Network(SPN)取代-重排網路

SPN是Claude Shannon在1949年提出，Claude Shannon認為一個好的加密方法必須有這兩種特色：

Diffusion 擴散 & Confusion 混亂

Diffusion: 是為了避免密文可以透過詞頻等分析被解出來，為了避免攻擊者可以透過語言中會出現的統計結構來破譯。同時為避免兩段類似文字的情形中被猜出加密方式與密鑰，也期望即便明文只有一些小變動也會使密文產生很大的變化。

Confusion: 是為了讓密文看起來更像隨機、無法被讀取，密文與金鑰間需夠複雜，不容易被突破。

SPN算法，其中主要有代換、置換和輪金鑰混合三個步驟：

代換：把明文的字母用另一個字母替換。

置換：調動字母的順序。

SPN想法是，單輪不夠的話，那就重複加密很多輪， 增加其複雜度。

Feistel Cipher
費斯妥加密

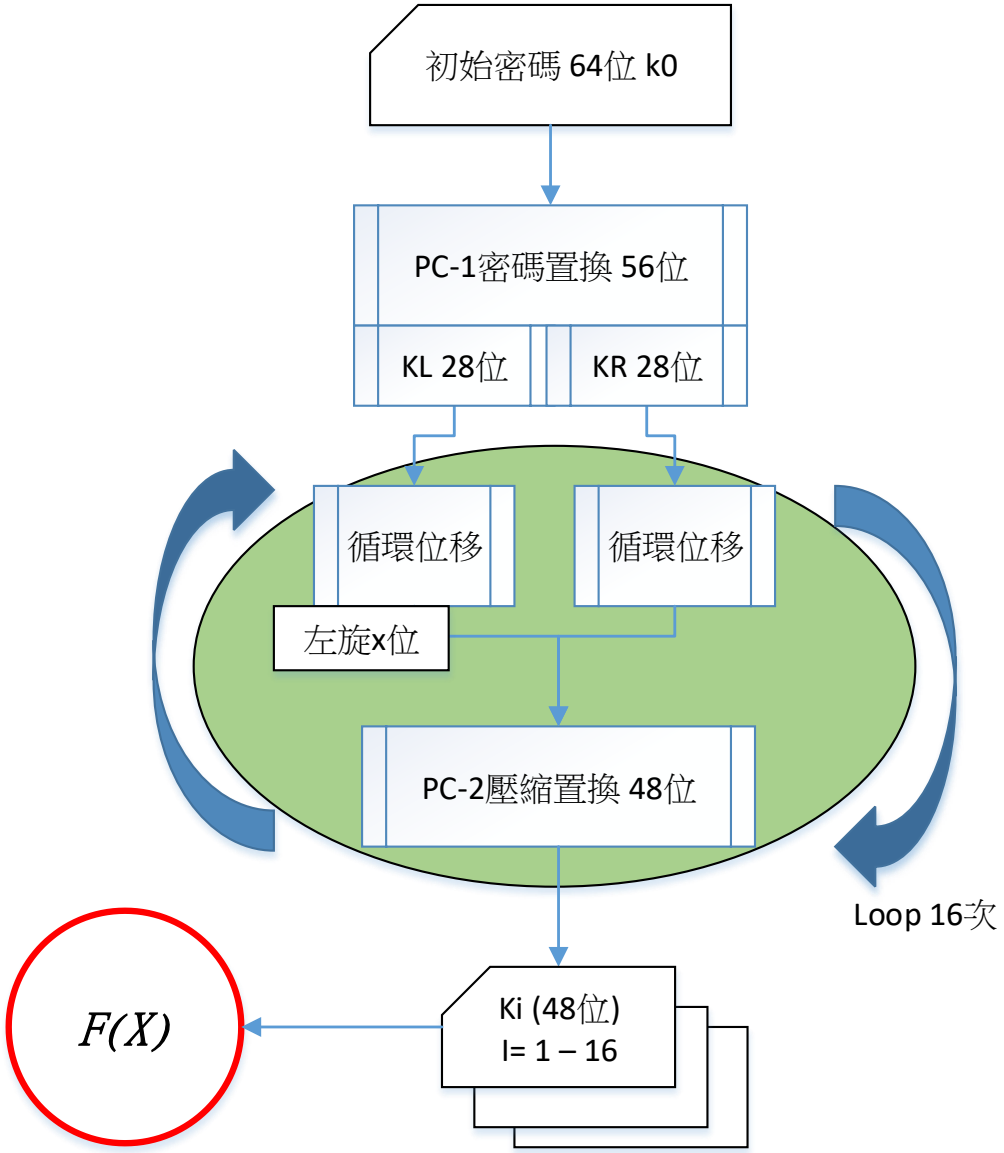
Step1: 產生 16把子金鑰

Step2: 明文加密

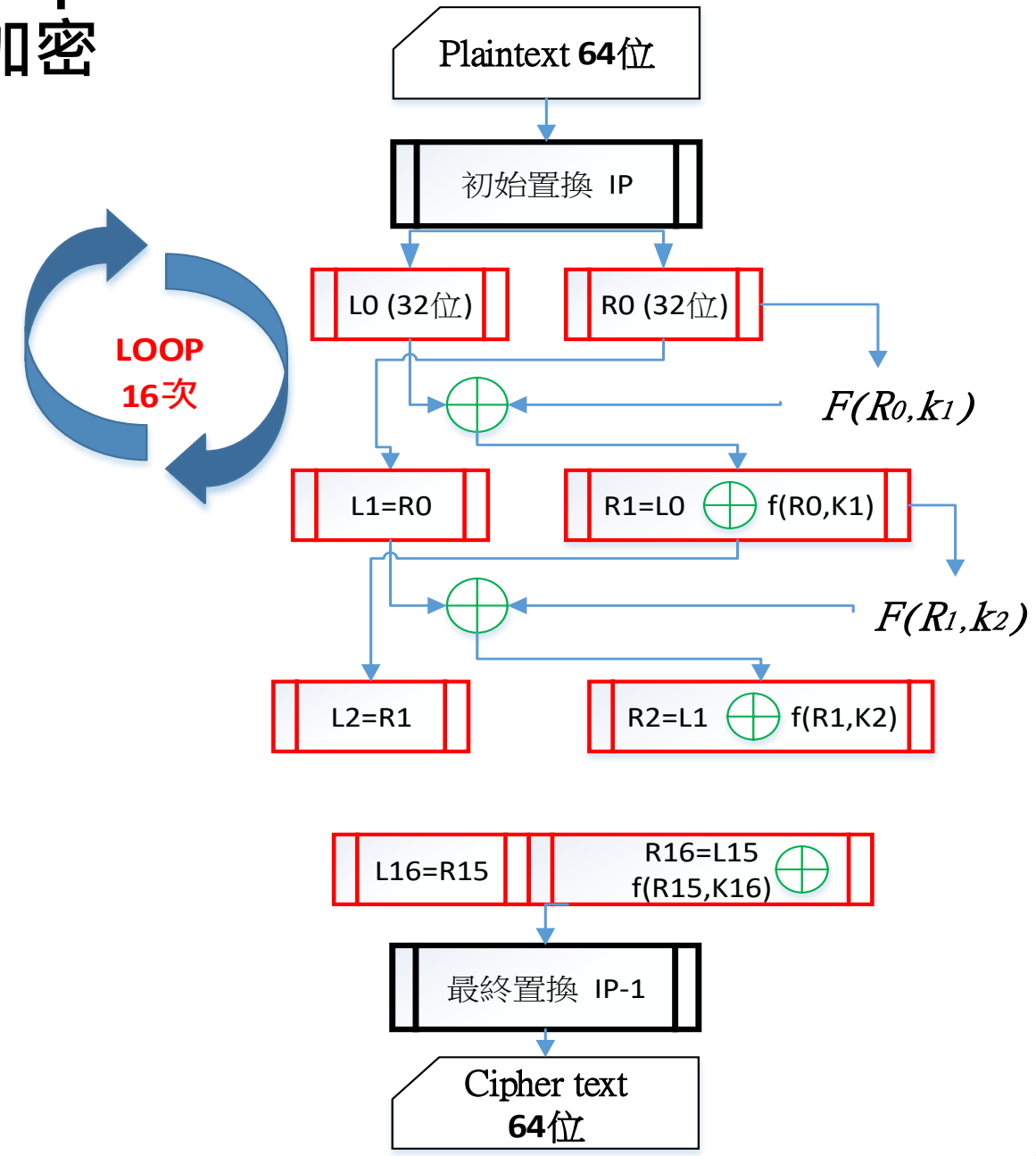
1) 64位元明文初始置換
2) 切分為 L/ R parts(32位元)
3) L0 XOR Feistel 函數之產出資料(input=R0 & K1)

   Set R1= [L0 XOR f(R0,K1)]; Set L1 = R0。
4) Loop 16次 ; R j+1= [L j XOR f(Rj ,Kj+1)]; L j+1 = Rj。
5) 合併 two parts L16 & R16
6) 最終置換

產生 子密碼 K1 – K16 for F 函數



Feistel Cipher
費斯妥加密

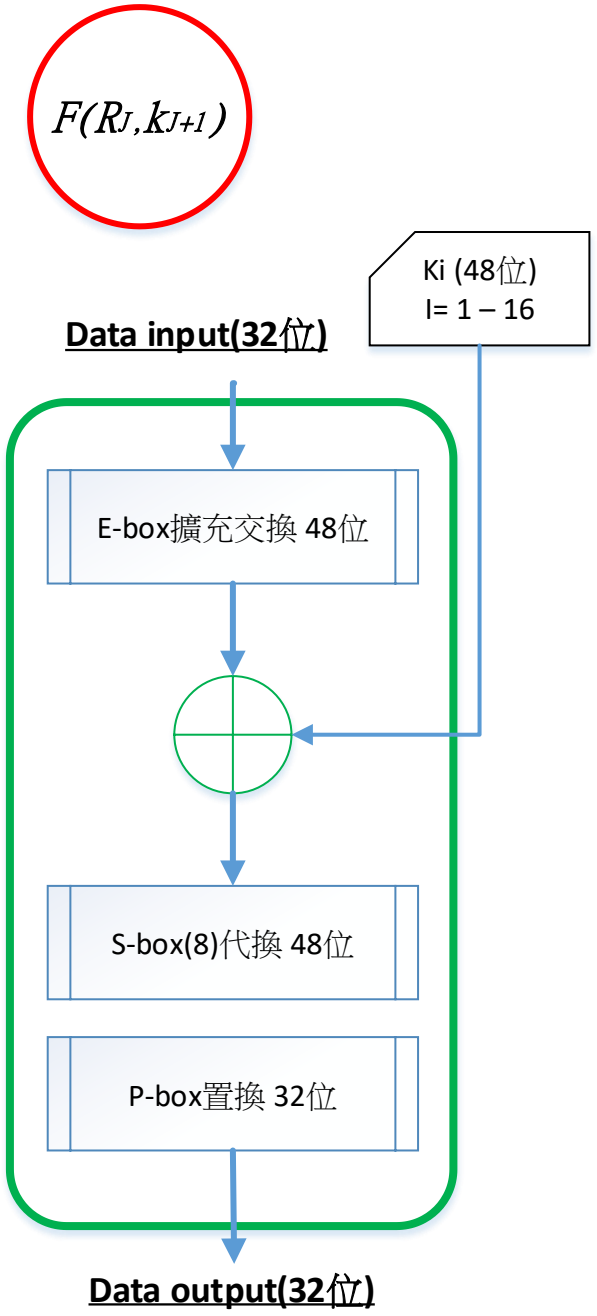


XOR Cipher

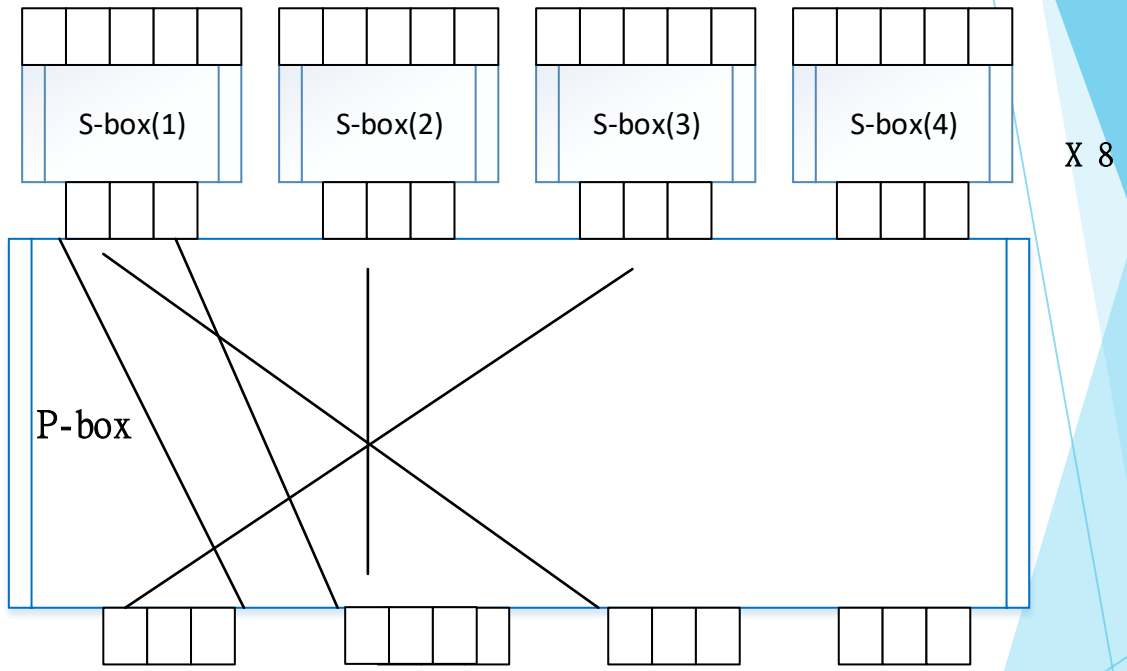
XOR加密法利用的就是任何一段文字被另一段文字XOR運算兩次後會得到原本的文字。

A⊕B=C C⊕B=A　0⊕0=0　1⊕1=0
1⊕0=1

A看作是要傳遞的明文、B是加解密用的金鑰、C是加密後的密文



48 位元切成 8個 parts



S-box= 4 * 16 array (contain from 0 -- 15) 0000-1111

First bit + last bit (2 bit) = row

Middle bit (4 bit) = column

TDES使用「金鑰包」，其包含3個DES金鑰，K_1，K_2和K_3

密文 = $E_{K3}(D_{K2}(E_{K1}(明文)))$

也就是說，使用K_1為金鑰進行DES加密，再用K_2為金鑰進行DES「解密」，最後以K_3進行DES加密。

而解密則為其反過程：明文 = $D_{K1}(E_{K2}(D_{K3}(密文)))$

每次加密操作都只處理64位元資料，稱為一塊。

3TDEA：

三個金鑰是獨立的。常用名稱為3TDEA或「三倍長度金鑰」（triple-length keys）

金鑰的強度最高，擁有3 x 56 = 168個獨立的金鑰位。

2TDEA：

K1和K2是獨立的，而K3=K1。常用名稱為2TDEA，或「雙倍長度金鑰」（double-length keys）安全性稍低，擁有2 x 56 = 112個獨立的金鑰位。

AES進階加密標準(Advanced Encryption Standard)

AES的區塊長度固定為128位元，金鑰長度則可以是128，192或256位元

AES加密過程是在一個4×4的位元組矩陣上運作，這個矩陣又稱為「體（state）」，其初值就是一個明文區塊（矩陣中一個元素大小就是明文區塊中的一個Byte 。加密時，各輪AES加密迴圈（除最後一輪外）均包含4個步驟：

AddRoundKey—矩陣中的每一個位元組都與該次回合金鑰（round key）做XOR運算；每個子金鑰由金鑰生成方案產生。

SubBytes—透過一個非線性的替換函式，用尋找表的方式把每個位元組替換成對應的位元組。

ShiftRows—將矩陣中的每個橫列進行循環式移位。

MixColumns—為了充分混合矩陣中各個直行的操作。這個步驟使用線性轉換來混合每行內的四個位元組。最後一個加密迴圈中省略MixColumns步驟，而以另一個AddRoundKey取代。

對稱密鑰演算法（Symmetric-key algorithm）

DES VS AES

2018年7月 NIST宣布 TDES正式退役，

2023年後TDES不允許使用在應用系統內。

	DES	AES
Developed	1977	2000
Key length	56 bits	128,192,256 bits
Cipher type	Symmetric block	Symmetric block
Block size	64 bits	128 bits
Security	Proven inadequate	Considered secure

非對稱密鑰演算法（Asymmetric-key algorithm）

公開金鑰加密演算法必須使用「私密金鑰(Private key)」與「公開金鑰(Public key)」，加密與解密使用不同的金鑰，主要使用RSA演算法、數位簽章演算法(DSA)、橢圓曲線密碼(ECC)三種演算法。

RSA加密演算法:

是一種非對稱加密演算法，在公開金鑰加密和電子商業中被廣泛使用。RSA是由羅納德·李維斯特（Ron Rivest）、阿迪·薩莫爾（Adi Shamir）和倫納德·阿德曼（Leonard Adleman）在1977年一起提出的。當時他們三人都在麻省理工學院工作。RSA 就是他們三人姓氏開頭字母拼在一起組成的。對極大整數做因數分解的難度決定了 RSA 演算法的可靠性。換言之，對一極大整數做因數分解愈困難，RSA 演算法愈可靠。

非對稱密鑰演算法（Asymmetric-key algorithm）

數位簽章演算法DSA（Digital Signature Algorithm）:

是框架在公鑰加密、模算數和離散對數演算法。該演算法使用由公鑰和私鑰組成的金鑰對。私鑰用於生成訊息的數位簽章，並且可以通過使用簽章者的相應公鑰來驗證這種簽章。

數位簽章提供資訊鑑定（接收者可以驗證訊息的來源），完整性（接收方可以驗證訊息自簽章以來未被修改）和不可否認性（傳送方不能錯誤地聲稱它們沒有簽署訊息）。僅供數位簽章使用，這一點與 RSA 演算法迥然不同，雖然 DSA 演算法不能使用於資料加密或交換鑰匙方面，但亦屬公開鑰匙演算法之一。

橢圓曲線密碼學（**Elliptic Curve Cryptography ECC**）

是一種基於橢圓曲線數學的公開密鑰加密演算法。

RSA金鑰計算方式

1. 選出兩個較大的質數 **p** 和 **q**
2. 計算兩個質數的乘積 N = p × q
3. 計算出小於 N 且與 N 互質的整數個數

   歐拉函數 r= **φ(N)= φ(p) × φ(q)** = (p - 1)(q - 1)
4. 選出一個整數 e （當做公鑰）選擇條件

   1 < e < **r**　　e 與 **r** 互質
5. 計算 d (私鑰) d 與 e 的關係

   (e × d) / r 餘數為 1　　K=1，2，3...時，e × d - k × r = 1

公鑰 **KU = {e, N}**　　私鑰 **KR = {d, N}**

RSA加解密方式

由於RSA 演算法式數值的運算，所以開始需要把資料字串化，使用ascii code、unicode code、utf-8編碼等將字串轉換為數字。轉換後數字X 需要小於密鑰組中 N 的長度，如長度大於N 則需要進行拆分。就是當加密資料太大需要先切割，分組處理的原因。

加密： X 的 e 次方 除以 N 求餘數 y ； X^e (mod N)= y

X 為明文轉化後的數字，**e為公鑰**，N 為整數(p * q)，y 為密文，mod 模除

解密： y 的 d 次方 除以 N 求餘數 X ； y^d (mod N)= X

y 為密文，**d 為私鑰** 此計算出餘數就可還原明文的X

非對稱密鑰演算法（Asymmetric-key algorithm）
數位簽章 & 簽章憑證 & PKI服務架構

於 PKI的機制下，當傳送方要傳送時，就用接收方的公鑰將訊息加密，接收方收到加密訊息後，再用私鑰解開，這樣即使有心人拿到公鑰，只要沒拿到接收方的私鑰，也還是無法解密訊息。但這還是有個問題，就是接收方要如何確認訊息真的是由傳送方傳的呢?假設有人在網路上找到了接收方的公鑰，假造一封有病毒的訊息，再用接收方公鑰加密傳過去，這樣接收方一打開不就中計了!

這時就需要數位簽章(Digital Signature)，也就是傳送方除了使用接收方的公鑰加密外，也使用傳送方自己的私鑰對該封加密訊息的Hash簽名(就是對該封訊息蓋上寄件者指印)，這樣當接收方收到時，除了要使用接收方本身的私鑰解密，也需要用寄件者的公鑰來對簽章作驗證，確認是由正確的傳送者傳來的。

若要建立數位簽章，需要有可識別身分的簽章憑證。 當您傳送有數位簽章的文件時，將一併傳送您的憑證及公開金鑰。 憑證由憑證授權單位發行(CA Certificate authority)，第三方可信任單位(如TWCA)。 憑證的有效期限通常是一年，之後，簽章者必須更新或取得新的簽章憑證來建立身分識別。

非對稱密鑰演算法（Asymmetric-key algorithm）數位簽章 & 簽章憑證 & PKI服務架構

公開金鑰基礎建設PKI服務架構

由三個單位所組成

- 憑證機構（CA）：

  公認且受信賴的服務提供者，確認公鑰持有者的身分，數位憑證的產生、簽發、廢止。

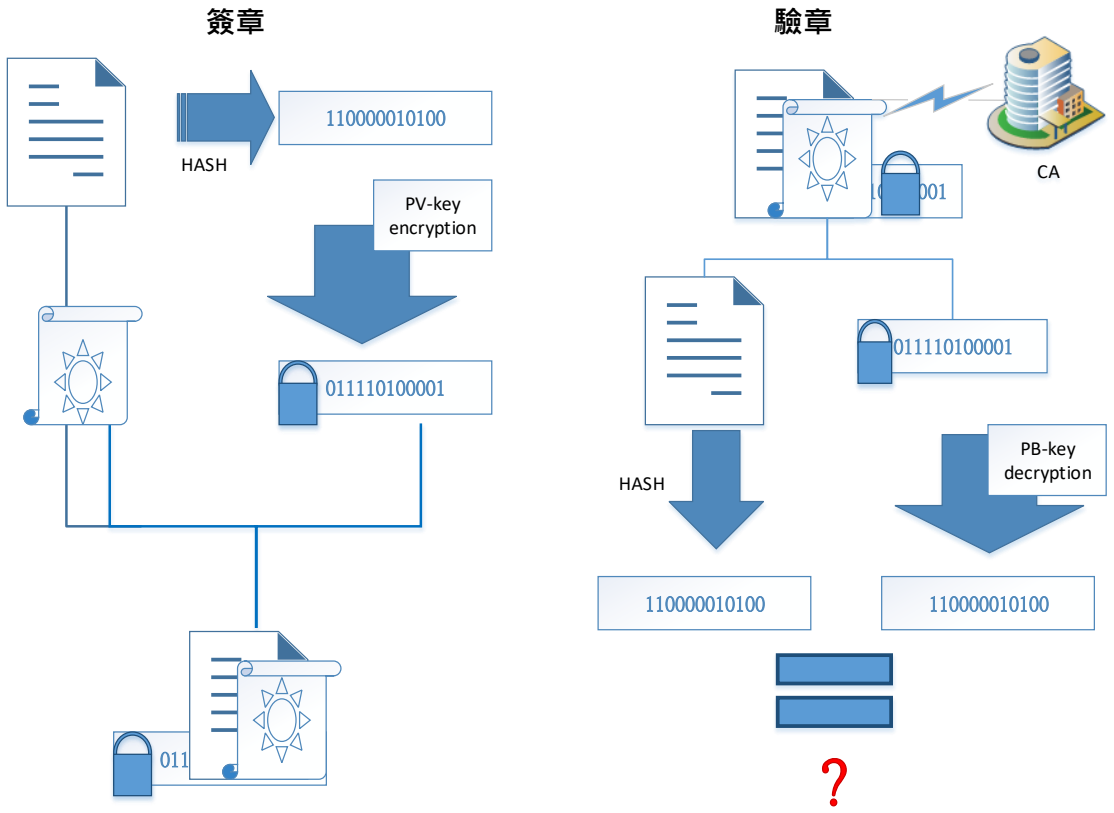
- 註冊管理中心（Registration Authority，RA）：

  驗證申請者的身分。

- 憑證廢止清冊儲存中心（Certificate Revocation List，CRL），CRL一般又稱作目錄服務（Directory Service，DS）：

  提供憑證狀況的查詢功能。

非對稱密鑰演算法（Asymmetric-key algorithm）
數位簽章驗證



- Step1：簽章憑證 送 CA驗證。**身分認證(Authentication)**。
- Step2: 接收方的私鑰解密文。**機密性(Confidentiality)**。
- Step3: 簽章驗證(HASH值)。**完整性(Integrity) &不可否認性(Non-Reputation)**。

公鑰: 可用於資料的加密(RCV-PB)與簽章的驗證(SND-PB)。

私鑰: 則用於資料解密(RCV-PV)與資料的簽章(SND-PV)。

銀行業務上的加密運用				
	TDES	RSA	MAC(message authentication code)	DS(Digital Signature & certificate)
財金跨行通匯	V		V	
財金跨行轉帳 & ATM	V		V(TAG) offset	
票交 EACH	V		V	
個網(網銀/行銀)	V	V(HTTPS)	V	
信用交易	V	V(device & card)	V ARQC (Authorisation Request Cryptogram) and ARPC) PVV CVV	
金融電子資料交換 (Financial Electronic Data Interchange，簡稱 FEDI		V		V
FXML (Financial eXtensible Markup Language) 亦稱金融XML		V		V
企網		V		V

財金與銀行間的 Keys

- CD-key(Cross-domain key) between bank & FISC

  BANK　　<- Paper <-　FISC

  CD-key交換 採用紙本 & A-part B-part寄給不同的收件者。

  A-part XOR B-part ➔ CD-key as KEK between bank & FISC

- Working key between bank & FISC

  BANK-A　　⬅====　　FISC

  [wk-key] cd-key-bank-A

  wk-key as MAC key between bank & FISC

- Working keys between Banks for 匯款

  BANK-A　　==➔　　FISC　　=➔　　BANK-B

  [S-key-A] cd-key-A　　[S-key-A] cd-key-B

  bank-a的sending-key, bank-b 以 receiving-key of bank-a存放於其 HSM上。

  每家銀行HSM上有 (N-1) * S-key + (N-1) * R-key。

  S-key/R-key as MAC key between banks

通匯是採 Store and Forward方式，FISC as a switch & settlement center

線上查驗

1. 通匯序號須 連續 & 唯一。
2. MAC check 1&2

   MAC check1= bank & FISC；

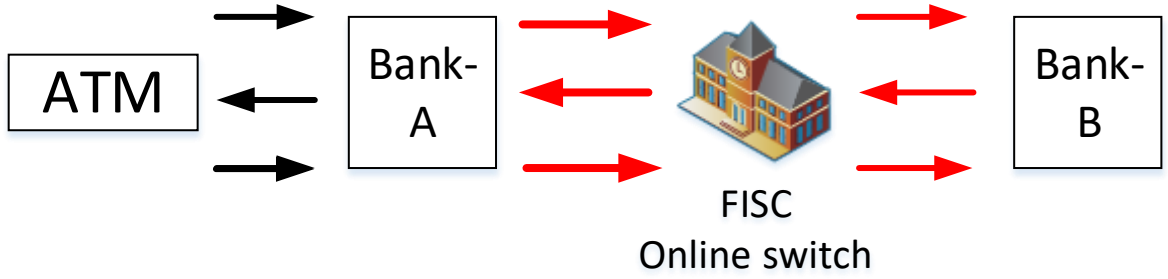
   MAC check2= bank-a & bank-b

密文 = [明文]mac-key-E

明文 = 部分交易資料內容

MAC= sub-string of 密文

ATM交易驗證



- 國內 ATM採 3ways 方式

1. ATM request to Host
2. Host response to ATM
3. ATM send confirm back to Host

ATM交易驗證

- 晶片金融卡 offline pin check on chip
  1. Pin offline check on chip - 確認持卡人

     Store offset on chip security area when pin change

     1) Offset= [pin block]pp-key-E ; Pin block= password + PAN(primary account number)。
     2) PP key(pin protection key)不提供 decipher，只提供 encipher & certificate。
  2. TAG online check on Host - 確認晶片
     1) Chip 產生 TAG(by key C6) as a MAC between chip & issue-bank
     2) 晶片金融卡 TAG as 信用卡 ARQC
     3) Chip key C6 - 每張晶片的 Key皆不同

        Derive by issue-bank-key IC6。[PAN]IC6-E ➔ C6 of chip。
  3. MAC check-確認資料

結論

密碼學：

- Symmetric-key algorithm
    1. 取代-重排網路(Substitution-Permutation Network) SPN
    2. 費斯妥加密(Feistel Cipher)
    3. TDES VS AES
- Asymmetric-key algorithm
    1. RSA金鑰計算方式
    2. 數位簽章； 簽章憑證
    3. 公開金鑰基礎建設PKI服務架構(CA; RA; DS)

銀行的加密運用 ：

1. 業務 mapping with TDES / RSA
2. Keys: CD key；Working key ; Key derive
3. Data check: MAC; ARQC；PWD certificate：PIN block

Q & A