國立交通大學

資訊工程系

博 士 論 文

代理簽章與前向預防式代理簽章之研究

Studies on Proxy Signatures with/without Proactive Property

研 究 生：陳以德

指導教授：葉義雄　教授

中 華 民 國 九 十 四 年 十二 月

代理簽章與前向預防式代理簽章之研究

Studies on Proxy Signatures with/without Proactive Property

研 究 生：陳以德　　Student：I-Te Chen

指導教授：葉義雄　　Advisor：Yi-Shiung Yeh

國 立 交 通 大 學
資 訊 工 程 系
博 士 論 文

A Dissertation
Submitted to Department of Computer Science
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy
in

Computer Science

December 2005

HsinChu, Taiwan, Republic of China

中 華 民 國 九 十 四 年 十二 月

國　立　交　通　大　學

博碩士論文全文電子檔著作權授權書

本授權書所授權之學位論文，為本人於國立交通大學 資訊工程 系，94 學年度第 1 學期取得博士學位之論文。

論文題目：代理簽章與前向預防式代理簽章之研究
Studies on Proxy Signatures with/without Proactive Property

指導教授：葉義雄

■ 同意　□不同意

本人茲將本著作，以非專屬、無償授權國立交通大學與台灣聯合大學系統圖書館：基於推動讀者間「資源共享、互惠合作」之理念，與回饋社會與學術研究之目的，國立交通大學及台灣聯合大學系統圖書館得不限地域、時間與次數，以紙本、光碟或數位化等各種方法收錄、重製與利用；於著作權法合理使用範圍內，讀者得進行線上檢索、閱覽、下載或列印。

論文全文上載網路公開之範圍及時間：

本校及台灣聯合大學系統區域網路	■中華民國 95 年 9 月 1 日公開
校外網際網路	■中華民國 96 年 9 月 1 日公開

授 權 人：陳 以 德

親筆簽名：_____

中華民國　94　年　12　月　30　日

國　立　交　通　大　學

博碩士紙本論文著作權授權書

本授權書所授權之學位論文，為本人於國立交通大學 資訊工程 系，94 學年度第 1 學期取得博士學位之論文。

論文題目：代理簽章與前向預防式代理簽章之研究
Studies on Proxy Signatures with/without Proactive Property

指導教授：葉義雄

■ 同意

本人茲將本著作，以非專屬、無償授權國立交通大學，基於推動讀者間「資源共享、互惠合作」之理念，與回饋社會與學術研究之目的，國立交通大學圖書館得以紙本收錄、重製與利用；於著作權法合理使用範圍內，讀者得進行閱覽或列印。

授 權 人：陳 以 德

親筆簽名：_____

中華民國　94　年　12　月　30　日

國家圖書館博碩士論文電子檔案上網授權書

ID:GT008617818

本授權書所授權之學位論文，為本人於國立交通大學 資訊工程 系，
94 學年度第 1 學期取得博士學位之論文。

論文題目：代理簽章與前向預防式代理簽章之研究
Studies on Proxy Signatures with/without Proactive Property

指導教授：葉義雄

茲同意將授權人擁有著作權之上列論文全文（含摘要），非專屬、無償授權國家圖書館，不限地域、時間與次數，以微縮、光碟或其他各種數位化方式將上列論文重製，並得將數位化之上列論文及論文電子檔以上載網路方式，提供讀者基於個人非營利性質之線上檢索、閱覽、下載或列印。

※ 讀者基於非營利性質之線上檢索、閱覽、下載或列印上列論文，應依著作權法相關規定辦理。

授 權 人：陳 以 德

親筆簽名：_____

中華民國　94　年　12　月　30　日

代理簽章與前向預防式代理簽章之研究

學生：陳以德　　指導教授：葉義雄博士

國立交通大學資訊工程系博士班

摘　　要

網際網路的發達，使得政府及工商業界的文書往來，漸漸地由紙式文件，改為利用網際網路傳遞的電子式文件；簽章部分也由傳統的印章改為電子簽章。因應此一電子化的趨勢，世界各國紛紛制訂電子簽章法來推行電子簽章；中華民國政府也於 2001 公告自 2002 開始施行電子簽章法。

電子簽章又稱數位簽章，其發展到 1996 年，Mambo 才提出代理簽章的概念。代理簽章提供了原始簽章者，可以授權給代理簽章者代簽電子簽章的功能，是近十年來，蓬勃發展的電子簽章應用之一。許多學者也提出增進代理簽章安全性及不同的代理簽章演算方法來實現代理簽章。但這些方法被質疑能否實際應用於現實生活，所以我們除了提出架構在 Quadratic Residues 上的代理簽章外；也建議代理簽章建構在標準的簽章法，如 DSA 及 ECDSA 等；並提出了建構在 DSA/ECDSA 的代理簽章法，藉由已充分討論過安全性的標準簽章法，使代理簽章成為現實可行的簽章機制。

為了解決金鑰曝光的問題，我們在現有的代理簽章法加入前向預防式 (proactive) 的概念，而提出了 proactive secret sharing proxy signature scheme。藉由定時更新金鑰的方式，確保了某一段時間內，簽章的安全性。proactive secret sharing proxy signature scheme 的復原機制，更可以在某一代理簽章者的 share 遺失或無法使用時，由其他的代理簽章者來復原其 share。

單向雜湊函數經常配合簽章使用來增進簽章的效率，自從王小雲教授提出在 2^{69} 的時間複雜度內可以找到單向雜湊函數 SHA-160 的碰撞後；我們也分析 SHA-160 的訊息處理模式，發現 SHA-160 有衰減(Decay)的現象，所以我們提出兩個改進 SHA-160 的訊息處理模式安全性的方法。期望我們對單向雜湊函數與代理簽章的分析與改進，能使電子簽章能實際地運用於日常生活中。

Studies on Proxy Signatures with/without Proactive Property

Student：I-Te Chen　　Advisor：Yi-Shiung Yeh

Department of Computer Science
College of Computer Science
National Chiao Tung University

Abstract

Due to the rapid progress of Internet, governments and enterprises change their paper-based documents to electronic ones, as well as hand-made signatures to digital signatures. The electronic signature relative regulations are established all over the world. Taiwan has also established the Electronic Signature Laws in 2001 and put into operation in 2002.

Mambo et al. are the first group who introduced the proxy signature scheme in 1996. The proxy signatures, with which the original signers can delegate their signing capability to the proxy signers, are the most popular application of digital signatures in the last decade. Lots of researchers proposed improvement or alternative mathematic base of proxy signatures without adopting Digital Signature Algorithm (DSA) or Elliptic Curve Digital Signature Algorithm (ECDSA); however, most of the proposed proxy signature schemes are not feasible in practice because their securities cannot be really proved. Therefore, we propose the proxy signature adopting DSA and ECDSA and firstly introduce Quadratic

Residues' concepts. Our scheme keeps not only the properties of the DSA/ECDSA but also fulfills the strong requirements of proxy signatures.

To solve key exposure problem, we adopt proactive concept into proxy signature and propose proactive secret sharing proxy signature scheme. The proactive secret sharing proxy signature scheme is based on verifiable secret sharing to against the active attacker. Consequently, the proactive secret sharing proxy signature scheme, which is a group-oriented scheme, provides the functionality of proxy signers' shares renewing and recovering.

One-way hash functions are important skills to make digital signatures efficient. Wang et al. reported their method to find a collision efficiently in SHA-160 within 2^{69} hash steps in February 2005. In fact, we can still discover the decay phenomenon with the application of a message schedule's judgment when inspecting how SHA-160 generates message schedule actually. Therefore, we would like to introduce two SHA-160 corrections to enhance the security of SHA-160. In general, we hope our enhancement of SHA-160 and new proxy signature schemes could be used in practice.

誌　　謝

感謝老師、學長、同學、學弟妹們的幫忙與支持，才使我能獲得這個博士學位；這一份榮耀將分享給所有協助、支持我的人們。首先感謝有著長者的風範的指導教授 – 葉義雄老師，在我碩、博士修業期間，給我學業上的指導及生活上的輔導；其學術成就及待人處世之道，均為我表率。其次感謝明信學長無私的協助，使我得以順利完成學位，是我畢生難得的益友。

感謝論文口試委員呂及人教授、孫宏民教授、黃士昆教授、詹進科教授、雷欽隆教授與蔡錫鈞教授(以上按姓氏筆劃排列)的深入指導與建議，對於我論文之教誨，也是我往後做學問隨時需警惕的原則。

感謝中華電信研究所，提供我與研究相關的工作機會，尤其是張耿豪經理、謝東明博士、王文正博士與景榮兄的提攜與照顧，在此誠心感謝。

我也將最多的榮耀分享給我父母、岳父母與妻子，感謝他們支持，使我得以安心進修，完成這個學位。僅以這一點名位，聊表無限的感激。感謝在美國的表姐及在法國的以禮幫我校稿；也感謝實驗室眾兄弟姐妹們(不論已畢業或在學中)的鼓勵與幫助，尤其是兩宇-定宇與鎮宇給我的幫忙；其餘無法一一表謝，在此一併致謝。

Contents

List of Tables

List of Figures

Chapter 1 Introduction

Due to the rapid progress of Internet, the evidence of possessing documents is especially important in the electronic world. The digital signature is developed to replace ordinary hand-written signatures without losing the properties of signer authenticity, data integrity and non-repudiation. Proxy signature scheme is one kind of digital signature applications. In this dissertation, we survey lots of proxy signature schemes and propose several novel proxy signature schemes. On the other hand, a one-way hash function is also an important skill to make digital signature efficient. Therefore, the security of one-way hash functions is also worth discussing in this dissertation.

1.1 Motivation and Related Work

When original signers cannot sign a document by themselves, they might delegate their signing capability to trustworthy proxy signers. For example, when the manager of a company will leave for the vacation, she/he needs to authorize her/his secretary to sign messages on behalf of her/him. To deliver manager's private key directly to her/his secretary is dangerous, nevertheless, the traditional digital signature does not provide functionality of proxy, either.

A proxy signature scheme was introduced by Mambo et al. [MUO96] to solve the proxy problem so that the original signer could delegate her/his signing capability to proxy signer without revealing her/his secret information. However, Mambo's scheme does not provide non-repudiation property [Zha97a][Sun99]; thus several papers propose non-repudiation proxy signature scheme [Zha97a][Sun99][HWW03][LHW98][LKK01b] which means both original and proxy signers cannot deny the signatures those are created exactly by themselves.

In addition, Mambo's proxy signature scheme is not a strong proxy signature scheme because it is not a proxy-protected signature scheme in which the original signer knows and can derive the proxy key on her/his own. On the contrary, in the proxy-protected proxy signature scheme, the original signer and proxy signer create the proxy key interactively so that the proxy signer can be protected from a malicious original signer. Hence, Lee and Kim [LK99][LKK01a][LKK01b] proposed the concept of the strong proxy signature, which defined the four requirements of the proxy signature: verifiability, strong unforgeability, strong identiability, and strong undeniability. The strong proxy signature should complete all the requirements of proxy signature.

In the first, most of proxy signatures are based on discrete logarithm problem [EIG85] including Mambo's one, so that Li, Tzeng and Hwang proposed generalization of proxy signature based on discrete logarithms [LTH03]. After that, Wu and Varadharajan proposed a proxy signature based on Chinese remainder theorem [WV99]. In 2002, Chen, Liu and Chung proposed a proxy-protected signature scheme based on elliptic curve cryptosystem [CLC02], then Hwang et al proposed generalization of proxy signature based on elliptic curves [HTT04]. Furthermore, Z. H. Shao proposed the proxy signature schemes based on factoring in 2002 [Shao02] and Qingshui Xue, Zhenfu Cao proposed "Factoring based proxy signature schemes," in 2005 [XC05]. It is desirable to design proxy signature schemes based on Quadratic Residues (QR) problem.

Fan and Lei proposed efficient blind signature scheme based on QR in 1996 [FL96] and improved their scheme in 1998 [FL98]. Therefore, by adopting Fan's signature scheme, we propose the proxy signature based on QR to provide another mathematical implement.

Unfortunately, most of the proposed proxy signature schemes prior to this date are not feasible in practice because the security of those schemes cannot be really proved without

adopting standard signature such as DSA/ECDSA. The Digital Signature Algorithm (DSA) based on ElGamal [EIG85] and Schnorr's [Sch90] signature schemes is a useful digital signature scheme and has become a U.S. Federal Information Process Standard (FIPS 186) in August, 1991; called as the Digital Signature Standard (DSS) [NIST00]. In addition, the Elliptic Curve Digital Signature Algorithm (ECDSA), a DSA reinforced by the Elliptic curve cryptosystems (ECC), was invented in 1985 [ANSI99], which was also accepted as a FIPS standard (FIPS 186-2) in 2000 [NIST00].

To conquer those disadvantages, therefore, we are the first one who propose proxy-protected signature scheme combining standard signature DSA/ECDSA, as well as the Public key infrastructure (PKI) mechanism [AF99][BPH02][CFSMW03], which are pretty well known by their security properties to reinforce the proxy signature in order to be used in practice.

In many applications, the security is assured whenever the secret key remains unrevealed; therefore, a proxy key exposure is also a serious problem for proxy signature schemes. Chang, Lin and Yeh proposed "Forward Secure Proxy Signature Scheme" in NCS 2003 to deal with the key exposure problem [CLY03]. In forward secure proxy signature scheme, the proxy signer renews her/his proxy keys and deletes the previous proxy keys periodically. Those deleted proxy keys cannot be recovered, needless to mention being revealed. In addition, many threshold proxy signature schemes are proposed in which the k out of n threshold schemes [DF89][Zha97b][KPW97][SLH99] [HWW03]. However, those threshold proxy signature schemes may be insufficient to construct a long-live scheme with the proactive properties to reinforce security and the proxy share cannot be recovery either.

The proactive secret sharing scheme [HJKY95], which is based on Verifiable Secret Sharing [Ped91], provides strong security for a secret sharing against the active attacker.

Consequently, the proactive secret sharing scheme is a verifiable group-oriented scheme, which provides shares renewing and recovery properties. Therefore, we adopt the concept of proactive to propose a proactive secret sharing proxy signature scheme.

A proactive secret sharing proxy signature could permit the shares of designated signers, called proxy signers, being renewed periodically without changing the secret. In particular, we apply the (t, n) threshold proxy signature scheme to allow any t or more then t signers to form a designated group from n proxy signers to sign messages on behalf of the original signer. The proxy shares of proposed scheme are periodically renewed; therefore, it will be hurtless even when the adversary obtains the proxy shares information in some period. In our proactive secret sharing proxy signature scheme; furthermore, one proxy signer can recover her/his own share from the other t proxy shares without revealing any information about the other proxy shares. Unless more than t other proxy signers cooperate and collude, the secret share algorithm is always secure.

Proxy blind signature scheme is a variant proxy signature scheme prior to this date [TLT02][SH04][LA05]. Blind signature allows a user receiving a given message signed by the original signer without revealing any information about the message itself. By using Schnorr blind signature, Tan et al. proposed two digital proxy blind signature schemes based on DLP and ECDLP in 2002 respectively [TLT02]. Moreover, Lal and Awasthi further pointed out that Tan et al.'s proxy blind signature schemes suffer from a kind of forgery attack and proposed a more efficient proxy blind signature scheme, which means Tan et al.'s schemes do not fulfill the unforgeability and unlinkability properties. Lal and Awasthi's scheme, however, does not satisfy the unlinkability property either. Therefore, Sun and Hsieh discuss the security of Tan and Lal's schemes in 2004 particularly [SH04].

Most documents are too large in size to sign digital signature; thus one-way hash

functions are important skills to make digital signature scheme efficient. SHA-160 is one of popular one-way hash functions and the security of SHA-160 is worth discussing. In 1998, F. Chabaud and A. Joux presented a method to find collisions in Secure Hash Algorithm (SHA)[NIST02] with 2^{61} time complexities [CJ98]. In 2004's crypto conference and in Feb. 2005, Wang et al. [WFLY05][WY05] developed efficient methods to find collisions in MD5, as well as in SHA-160 with time complexity of 2^{39} and 2^{69} hash steps respectively. Furthermore, Biham and Chen [BC04] announced new analytical discoveries concerning SHA-160. Their results include a collision in a reduced-round version of SHA-160, which can be found less than 40 rounds.

Suppose the output size of one-way hash function is *n*-bit. According to the birthday paradox attack property [MOV96], we could expect certain collisions after trying $2^{n/2}$ possible input values. Van Oorschot and Wiener [OW94] have explained how such a brute-force attack might be implemented. That implies any cryptanalysis method with higher complexity than the birthday paradox attack will be regarded as inefficient. F. Chabaud and A. Joux find collision in SHA with 2^{61} complexities, related to differential cryptanalysis of block ciphers [CJ98], and their method is theoretically faster than birthday paradox attack. Unfortunately, in SHA-160, their method is unable to detect collision faster than the birthday paradox attack.

In fact, we can still discover the decay phenomenon with the application of a message schedule's judgment when inspecting how SHA-160 generates message schedule actually. Furthermore, we find a reason why move SHA to SHA-160. The more nonlinear terms are involved, the more terms in message schedule process will be effective. Therefore, we would like to introduce two SHA-160 corrections to enhance the security of SHA-160. This analysis could also be used in all SHA-serials or other one-way hash functions.

1.2 Organization of Dissertation

We describe our motivation and related work in this chapter; and then report some fundamental cryptosystem knowledge and discuss one-way hash functions in chapter 2. In chapter 3, we describe some preliminaries of digital signature, (strong) proxy signature, Quadratic Residues, secret sharing, DSA, ECDSA, etc. In chapter 4, we propose novel proxy signatures based on QR, DSA and ECDSA respectively and analyze security of proposed proxy signature schemes. And then, we proposed a proactive secret sharing proxy signature in chapter 5 to deal with key exposure and key recovery problems. Finally, we summarize a conclusion in chapter 6 and list references respectively.

Chapter 2 Cryptography

2.1 Cryptosystem

A cryptosystem can provide following properties [Sta03]: 1. **Secrecy:** It can prohibit the eavesdroppers from receiving plaintext. 2. **Authentication:** It can identify a message from its origin for the receiver and the eavesdroppers cannot disguise as someone. 3. **Integrity:** It can verify that a message has not been modified so that eavesdroppers can't replace a legal message by a false one in transmission. 4. **Non-repudiation:** It can prove the message of the sender who may falsely deny later that he had sent the message. And a cryptosystem is composed of five basic components:

m : plaintext message space.
c : ciphertext message space.
K : key space.
E : Encryption.
D : Decryption.

We show mathematic form and figure as follows:

$E_{k_e}(m) = c$ For a given key $k \in K$

$D_{k_d}(c) = m$ For a given key $k \in K$

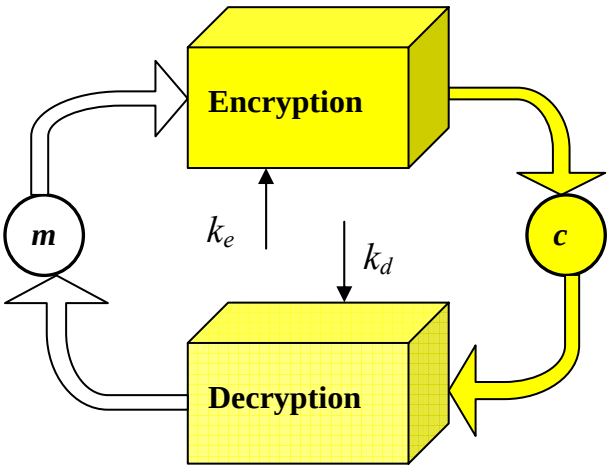

Fig 2.1 Encryption and Decryption

In Fig 2.1, cryptosystem uses keys k_e and k_d for **Encryption** and **Decryption** respectively. Simmons [Sim79] classifies the cryptosystems as symmetric (one key) and asymmetric (two keys). In symmetric cryptosystem, also called secret-key cryptosystem, the encryption key and the decryption key are the same or can be easily determined from each other. On the other hand, in asymmetric cryptosystem, also called public-key cryptosystem, the encryption key and the decryption key are different.

In Kerckhoffs's assumption [Ker83], the strength of a cryptographic system cannot rely on attacker's unawareness about the cryptosystem algorithm. A secure cryptographic system must be published and unbreakable even under the most fatal attack by the world's best cryptographers for years.

2.2 Symmetric Ciphers

There are two kinds of symmetric ciphers, stream ciphers and block ciphers. The stream ciphers encrypt plaintext one byte or one bit in one time span; one-time pad is one kind of stream ciphers. On the other hand, the block ciphers operate on fixed-length groups of bits to form the blocks.

2.2.1 Encryption Standard in U.S.

FIPS (Federal Information Processing Standard) - 46: DES (Data Encryption Standard) announced by U.S. Government in 1977 has been generally used. DES is a 64-bit block cipher with the key length of 56 bits. Unfortunately, Electronic Frontier Foundation (EFF) using a special purpose "DES cracker" machine proved DES insecure in July 1998 [EFF98]. Therefore, NIST (National Institute of Standards and Technology) announced Triple DES as FIPS 46-3 to enhance original DES. Triple DES uses three keys and three executions of the DES algorithm following an encrypt-decrypt-encrypt sequence.

Triple DES with 168 bits key and 64 bits the same block size as DES is not for long-term use [Sta03]. For reasons of both efficiency and security, a larger block size is desirable. Hence, NIST began the process of replacing DES with AES (Advanced Encryption Standard) in 1997 and Rijndael was published as AES: FIPS – 197 in 2001 [NIST01]. AES uses a 128 bits block size and its key length that can be 128, 192, and 256 bits. Four different stages are used in AES: substitute bytes (S-box), shift rows, mix columns, and add round key.

2.2.2 Dynamic Extended DES

The original S-boxes of DES are important design to resist differential attack. Furthermore, Yeh and Hsu proposed the extended DES [YH02], which developed eight more new S-boxes with the same cryptographic properties as original S-boxes in DES. These 16 S-boxes are used to construct the extended DES, which double the block cipher and key size. As a result, the time complexity of differential cryptanalysis of the extended

DES is 2110. We propose an intricate extended DES that includes permutation on S-boxes. By keeping the permutation information in secret, the new version of extended DES is stronger to defeat differential and linear attacks by 20922789888000 times.

The Extended DES

The extended DES [YH02] has exactly the same data flow and concept as DES. The eight more S-boxes are used in the extended DES to double the block cipher and key size. Some modifications are necessary on P-box and key scheduling algorithms.

The extended DES encrypts a 128-bit data block with a 112-bit key. All data bits go through an initial permutation. The data bits then split into two 64-bit data blocks called as right and left data blocks. Two data blocks then go through 32 identical rounds, there is no swap of two data blocks in the last round. After the last round, two data blocks are combined into a 128-bit block. The result will be through the inverse initial permutation.

In each round, the right data block and 96-bit sub-key (R_i and K_i in Figure 2.2) are combined by a round function called F. The output of F is then combined with the left part data block by XOR operation. The two data blocks swap in the next round.

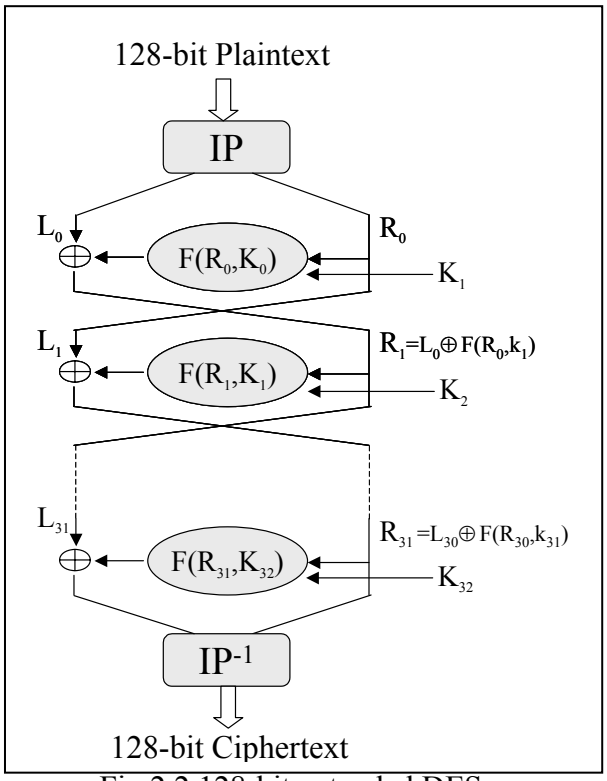

Fig 2.2 128-bit extended DES

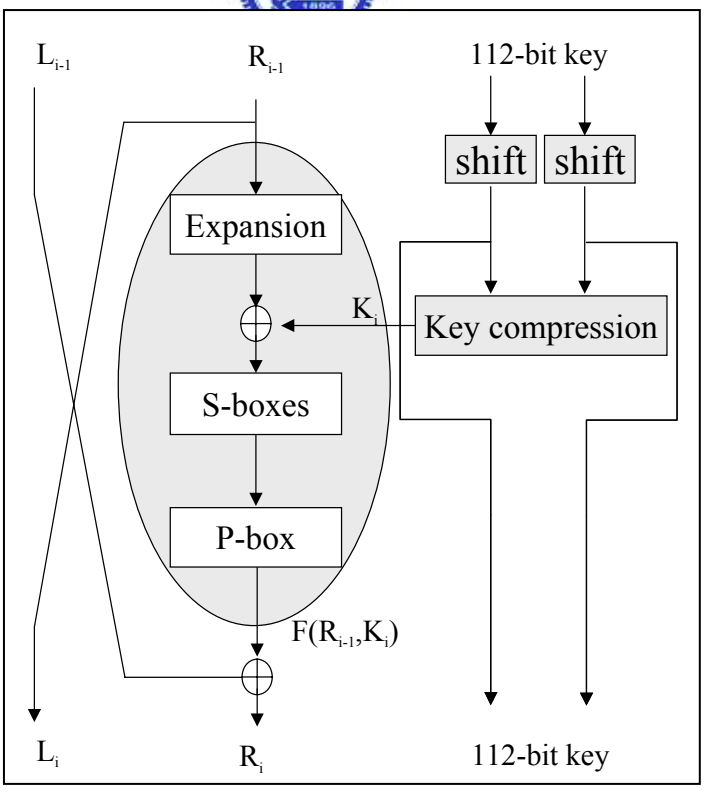

Fig 2.3 One round of 128-bit extended DES

The 64-bit right data block is expanded to 96 bits by expansion permutation after combining with the 96-bit sub-key; the 96-bit data is distributed to all 16 S-boxes as input. Each S-box has 4 output bits. Therefore, 64-bit data is used in the next step where P-box is the permutation box.

Eight more new S-boxes are proposed in following tables. Table 2.1 shows the cryptographically similarity of new S-boxes and original S-boxes. They are also semi-similar. The new S-boxes are listed in Table 2.2.

Table 2.1 The similarity of new and original S-boxes.

New design	Original	LST	B1	B2	C order	GD	ID	OD	L1	L2	L3	L4	GL	None-zero rate
S-box #9	S-box #1	20	3	3	1	9.31	32.25	46.56	18	20	22	18	78	79.4%
S-box #10	S-box #2	28	3	3	1	11.22	35.81	56.32	22	20	18	18	78	78.6%
S-box #11	S-box #3	24	3	4	1	12.65	41.70	63.62	18	22	20	18	78	79.6%
S-box #12	S-box #4	12	3	2	2	8.16	32.66	44.00	22	22	22	22	88	68.5%
S-box #13	S-box #5	20	3	2	1	9.90	35.81	55.32	22	20	18	20	80	76.5%
S-box #14	S-box #6	24	3	3	1	11.31	38.85	59.53	20	20	20	20	80	80.4%
S-box #15	S-box #7	24	3	3	1	12.17	43.45	65.18	18	22	14	20	74	77.2%
S-box #16	S-box #8	20	3	2	1	10.95	38.71	56.21	22	20	20	22	84	77.1%

LST: Linear structure tolerance.
B1: First order 0-1 balance tolerance.
B2: Second order 0-1 balance tolerance.
C order: Maximum order of completeness.
GD: Global SAC-map distance.
ID: Input SAC-map distance
OD: Output SAC-map distance
Li: Nonlinearity of output bit i.
GL: Global nonlinearity
None-zero rate: Percentage of none zero entry in the DDT map.

Table 2.2 Extended S-boxes.

3 0 9 7 15 12 6 11 14 13 2 1 5 10 8 4 0 3 5 8 9 15 12 6 13 10 11 7 14 4 2 1 15 5 12 2 0 11 9 14 4 3 1 8 10 6 7 13 9 15 0 5 10 6 3 8 2 12 13 11 4 1 14 7 S-box #9	1 10 15 12 8 3 6 5 13 4 0 7 14 9 11 2 4 7 10 0 15 9 1 12 8 14 3 13 5 2 6 11 2 5 4 10 7 12 9 3 11 8 14 1 13 6 0 15 7 0 9 3 4 15 10 6 2 13 5 14 11 8 12 1 S-box #10
15 4 12 1 5 10 2 13 3 8 6 11 0 7 9 14 6 13 15 2 8 4 5 11 0 7 9 12 3 10 14 1 4 13 15 10 2 1 8 6 14 3 0 5 11 12 7 9 13 3 1 4 11 14 2 8 7 10 12 15 0 5 9 6 S-box #11	10 7 15 12 4 2 1 11 0 13 5 3 9 14 6 8 6 13 12 0 1 7 11 14 3 8 9 15 10 4 5 2 4 1 2 11 15 12 8 6 7 10 14 5 0 9 13 3 1 11 7 14 12 0 2 5 13 6 4 9 3 10 8 15 S-box #12
4 7 1 12 14 11 8 2 13 10 6 9 0 5 3 15 13 0 2 7 4 14 1 11 3 12 5 10 15 9 8 6 10 1 12 11 9 2 7 14 6 13 15 4 5 8 0 3 7 11 9 4 2 1 14 13 0 6 10 3 12 15 5 8 S-box #13	2 14 15 0 12 11 9 5 4 13 8 3 1 6 7 10 12 5 9 10 7 0 2 15 3 6 14 13 8 11 4 1 12 2 3 14 15 4 10 9 11 1 5 8 6 13 0 7 1 15 12 5 10 9 7 2 6 8 0 14 3 4 13 11 S-box #14
13 2 4 7 3 12 8 1 0 15 14 9 5 10 11 6 3 8 14 13 9 2 5 11 15 4 0 10 12 7 6 1 2 11 8 13 15 0 4 14 12 5 1 6 10 3 7 9 13 6 1 8 2 11 14 5 10 9 12 3 7 4 0 15 S-box #14	12 2 10 7 1 4 15 8 11 14 0 9 13 3 6 5 2 1 9 4 7 14 12 11 13 8 3 15 10 5 0 6 1 11 15 8 4 13 2 7 14 0 5 6 3 10 9 12 13 6 1 8 2 5 14 4 7 10 12 15 9 3 0 S-box #16

Permuted S-boxes

Extended DES has 16 fixed S-boxes, each of them is a mapping from {0, ...,63} to {0, …,15}, or formulated as *S*: [0...63]→[0...15], used in a settled order. Unfortunately, this usage is convenient for cryptanalysis. To remedy the situation, more complicated use of S-boxes should be effectuated.

The change is to rearrange the order of S-boxes in the succeeding round. In detail, a permutation mappings p: [1...16]→[1...16] is used to construct the new order. The i^{th} S-box in the j^{th} round will be equal to the $p(i)^{th}$ S-box in the $(j\text{-}1)^{th}$ round. For example, the S-boxes sequence in the former round is $S_1\ S_2\ S_3\ S_4\ S_5\ S_6\ S_7\ S_8\ S_9\ S_{10}\ S_{11}\ S_{12}\ S_{13}\ S_{14}\ S_{15}$

S_{16} and given the permutation as (3,9,16,2,11,7,10,8,1,12,4,14,6,13,5,15); in the next round, the S-boxes sequence then becomes $S_3\ S_9\ S_{16}\ S_2\ S_{11}\ S_7\ S_{10}\ S_8\ S_1\ S_{12}\ S_4\ S_{14}\ S_6\ S_5\ S_{15}$.

By keeping the permutation information in secret, the exact usage of S-boxes is not explicit. This increases the difficulty of cryptanalysis.

Substitution Words Access

The whole S-boxes data can be filled into a table that forms as a two-dimension, 16x64, matrix. Without loss of generality, let the table be M[1...16, 1...64] and the initial S-boxes sequence be $S_1\ S_2\ S_3\ S_4\ S_5\ S_6\ S_7\ S_8\ S_9\ S_{10}\ S_{11}\ S_{12}\ S_{13}\ S_{14}\ S_{15}\ S_{16}$. The k^{th} word (4-bit) of S_i is placed in M[i, k]. While applying an S-boxes permutation p, the S-boxes sequence of first encrypting round will be $S_{p(1)}\ S_{p(2)})\ Sp_{(3)}\ S_{p(4)}\ S_{p(5)}\ S_{p(6)}\ S_{p(7)}\ S_{p(8)}\ S_{p\,(9)}\ S_{p\,(10)}\ S_{p\,(11)}\ S_{p\,(12)}\ S_{p\,(13)}\ S_{p\,(14)}\ S_{p\,(15)}\ S_{p\,(16)}$; that is, the k^{th} word of the i^{th} S-box is placed in M[p(i), k] now. Generally, the S-boxes sequence of the j^{th} round is:

$$S_{p^j(1)}S_{p^j(2)}S_{p^j(3)}S_{p^j(4)}S_{p^j(5)}S_{p^j(6)}S_{p^j(7)}S_{p^j(8)}S_{p^j(9)}S_{p^j(10)}S_{p^j(11)}S_{p^j(12)}S_{p^j(13)}S_{p^j(14)}S_{p^j(15)}S_{p^j(16)},$$

where $p^j(i)$ denotes to execute the mapping p with j times such as p(p(...p(p(i))...)). It is obviously that the k^{th} word of the i^{th} S-box of the j^{th} round is placed in M[$p^j(i)$, k].

According to the above derivation, we know that a word in an S-box can still be easily read from the S-boxes table while including the S-boxes permutation. The increasing calculations are just some mapping operations and never exceed 16 times of nested mapping because of the restriction of 16 rounds in extended-DES. Therefore, the new algorithm is considered as the same efficient as extended-DES. While decrypting, the same 16 S-boxes sequences in encryption are used but in reverse order. This does not increase the computing time complexity.

Permutation Materials

The adopted S-boxes permutation should be kept in secret. It can be added up some other secret information like an independent key to the system. This will increases the quantity of secret information; system will be more secure in this viewpoint. On the other hand, there turns out more secret data to be managed which may raise the burden for users.

Alternatively, the S-boxes permutation can be also derived from the key. For example, we can choose the smallest integer between A and B, which is larger than the key value and relatively prime to 16 as the multiplier. The i^{th} value of the permutation function p, will be p(i)=(A+i*B mod 16)+1.

Security Analysis on Dynamic Extended DES

Both differential and linear attacks need to know the exact usage of S-boxes. If we can keep the permutation in secret, it will be difficult for the adversary to apply the two attacks. The attack may guess the permutation with rare $\frac{1}{20922789888000}$ probability because 16 S-boxes can derive 16! = 20922789888000 different permutations. It is computational inefficiency to guess the right permutation.

Furthermore, if higher security is required, the permutations used in each round can be different. That is, using 16 different permutations to construct the initial S-boxes sequence, and applying them in different rounds. The probability to guess the right permutation reduces to $\frac{1}{20922789888000^{16}} \cong \frac{1}{1.34869 \times 10^{214}}$ to be computational impossible.

Dynamic Extended DES permutes the S-boxes order in the succeeding round; as a result, the usage of S-boxes becomes more confused. This change can enhance extended DES to resist differential and linear attacks. In addition, this method can be also used in

any other S-boxes. However, the permutation information should be always kept in secret; otherwise, not only the confusion effect will no more exists, but also become even favorable for the cryptanalysis.

2.2.3 NESSIE

New European Schemes for Signature, Integrity, and Encryption (NESSIE) project has launched out the next generation of cryptographic algorithms in 2000 [Nessie04]. The NESSIE portfolio of cryptographic primitives has been announced in February 2003. In block cipher scheme, MISTY1 (64-bit), AES (128-bit), Camellia (128-bit), SHACAL-2 (256-bit) are recommended algorithms. MISTY1 is similar to the block cipher KASUMI, which has been scrutinized prior to its adoption as a 3GPP standard, so many analyses for KASUMI would be also applicable to MISTY1. AES is FIPS – 197 announced by U.S. NIST; and Camellia has many similarities to the AES. SHACAL-2 is based on a one-way hash function upon SHA [NIST02] used in encryption mode. The strength of SHACAL-2 is inheritance from the extensive analysis that has been made on SHA. Although RC6 is also a secure block cipher, the NESSIE felt unable to consider RC6 [RRSY98] owing to ongoing serious Intellectual Property Rights issues.

2.3 Asymmetric Ciphers

Diffie Whitfield and Hellman introduced asymmetric ciphers in 1976 [DH76]. Asymmetric ciphers rely on one key for encryption and a different but related key for decryption; nevertheless, it is computationally infeasible to determine the decryption key given only the knowledge of the algorithm and encryption key. For example, asymmetric ciphers cryptosystem encrypting the sender's messages by using recipient's "public" key

and the recipient's "private" key can decrypt the messages. RSA [RSA78], ElGamal [ElG85] and Elliptic curve [Men93][ANSI99][Han04] are most popular asymmetric cryptosystems that we describe as follows:

2.3.1 RSA Cryptosystem

Rivest, Shamir, and Adleman developed the RSA algorithm in 1977 [RSA78]; the letters RSA are the initials of their surnames. The RSA scheme makes use of factoring problem to generate key pairs described as follows:

1. Let p and q are large primes such that $p \neq q$ and $n = pq$.
2. Compute the Euler's totient function $\phi(n) = (p-1)(q-1)$.
3. Choose a integer e, where $1 < e < \phi(n)$ and e is coprime to $\phi(n)$ i.e. gcd $(e, \phi(n))=1$.
4. Compute d such that $ed = 1 \bmod \phi(n)$.
5. The public key is (e, n) and the private key is (d, n).

Euler's theorem shows that $\alpha^{\phi(n)} \bmod n = 1$; thus to encrypt message m, we could compute $m^e \bmod n = c$ to obtain ciphertext c. And to decrypt ciphertext c, we could compute as follows:

$$
\begin{aligned}
& c^d \bmod n \\
&= (m^e \bmod n) \bmod n \\
&= m^{ed} \bmod n \\
&= m^{(k\phi(n)+1)} \bmod n \\
&= m^1 \bmod n \\
&= m
\end{aligned}
$$

We use artificially small parameters here; but we can also use OpenSSL [OpenSSL] to generate and examine key pairs. For example, let p = 101, q = 53, n = 101*53 = 5353, the message m = 4657, and choose e = 743 for public key. Via Euclidean algorithm, we could compute the private key d = 7 so that the public and the private key are (743, 5353) and (7, 5353) respectively. The encryption function is

$$E_{743}(4657) = 4657^{743} \bmod 5353 = 1003$$

and the decryption function is:

$$D_7(1003) = 1003^7 \bmod 5353 = 4657$$

Both of these computations can be done efficiently using the square-and-multiply algorithm for modular exponentiation. RSA is much slower than symmetric cryptosystems. In practice, sender typically encrypts a secret message with a symmetric algorithm, encrypts the symmetric key with RSA, and transmits both the RSA-encrypted symmetric key and the symmetrically- encrypted message to receiver.

2.3.2 Discrete Logarithm problem

Discrete logarithms are defined in group theory, which is a collection of elements together with a binary operation. A primitive root $g \in$ a finite group $\boldsymbol{Z_p}^*$, a number x under multiplication modulo the prime p, and g^x denoting the element obtained by multiplying g itself by x times; by Fermat's little theorem, we know that for a primitive root $g \in \boldsymbol{Z_p}^*$, $g^p = g \bmod p$, and the set of group is:

$$\{g, g^2, g^3, \ldots, g^{p-1}\} = \{1, \ldots, p-1\}.$$

The discrete logarithm problem is as follows: given a primitive root g in $\boldsymbol{Z_p}^*$ and another element $y \in \boldsymbol{Z_p}^*$, finding an integer x such that $g^x = y \bmod p$. For example, the solution to the problem $3^x = 15 \pmod{17}$ is 6, because $3^6 = 729 = 15 \pmod{17}$. If in $\boldsymbol{Z_n}^*$, where n is not a prime number, by Euler's theorem, $\alpha^{\phi(n)} \bmod n = 1$, the set of group will be:

$$\{\alpha, \alpha^2, \alpha^3, \ldots, \alpha^{\phi(n)}\}.$$

The ElGamal cryptosystem [ElG85] is based on the discrete logarithm problem. For a generator (primitive root) $g \in \boldsymbol{Z_p}^*$ of order p, Alice chooses a random x from $\{0, \ldots, p-1\}$ and computes $y = g^x$. The values p, g, and y are the public keys and x is a private key of Alice. To encrypt a message m to Alice, we show as follows:

1. Convert m to into an element of $\boldsymbol{Z_p}^*$
2. Choose a random k from $\{0, \ldots, p-1\}$
3. Calculate $c_1 = g^k \bmod p$ and $c_2 = my^k \bmod p$
4. Send the ciphertext (c_1, c_2) to Alice

Then, Alice can decrypt ciphertext by computing $c_2 (c_1^x)^{-1}$.

$$c_2 (c_1^x)^{-1} = my^k (g^{kx})^{-1} = mg^{xk} (g^{kx})^{-1} = m \bmod p.$$

2.3.3 Description of Elliptic Curves

In general, elliptic curves take the form: $y^2 + axy + by = x^3 + cx^2 + dx + e$ where a, b, c, d, and e are the real numbers satisfying to some conditions [Han04]. There are two finite fields $\boldsymbol{Z_p}^*$ and $Z_{2^n}{}^*$. The elliptic curve E over $\boldsymbol{Z_p}^*$ and $Z_{2^n}{}^*$ are defined as definition 2.1 and 2.2 respectively:

***Definition* 2.1**: Let $a, b \in \boldsymbol{Z_p}^*$ be constants such that $4a^3+27b^2 \neq 0$. An elliptic curve is the set E of solutions $(x, y) \in \boldsymbol{Z_p}^*$, to the equation:

$$y^2 = x^3 + ax + b \tag{2.1}$$

together with a special point O called the point at infinity.

***Definition* 2.2**: Let $a, b \in Z_{2^n}{}^*$ be constants such that $b \neq 0$. An elliptic curve is the set E of solutions $(x, y) \in Z_{2^n}{}^*$, to the equation:

$$y^2 + xy = x^3 + ax^2 + b \tag{2.2}$$

together with a special point O called the point at infinity.

We concentrate on elliptic curves over finite fields $\boldsymbol{Z_p}^*$. An example of elliptic curve E over $\boldsymbol{Z_p}^*$ as following:

Let $p = 19$ and consider the elliptic curve E: $y^2 = x^3 + x + 4$ defined over $\boldsymbol{Z_{19}}^*$. In this case, $a = 1$ and $b = 4$. We have $4*1^3+27*4^2 \pmod{19} = 18 \neq 0$, which satisfies the condition for an elliptic group mod 19. The order of points in $E(\boldsymbol{Z_{19}}^*)$ is also 19 and all the points and O are list as following:

Table 2.3 Points on the Elliptic Curve $E(\boldsymbol{Z}_{19}{}^*)$

(0, 2)	(6,13)	(11, 4)
(0,17)	(8, 7)	(11,15)
(1, 5)	(8,12)	(14, 8)
(1,14)	(9, 1)	(14,11)
(5, 1)	(9,18)	*O*
(5,18)	(10, 8)	
(6, 6)	(10,11)	

The addition and multiplication operation in ECC are counterpart of modular multiplication and exponentiation in RSA, respectively. Let $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ be two points on an elliptic curve E. Then, $P + Q = R$, we show it as Fig 2.4 and Fig 2.5 geometrically.

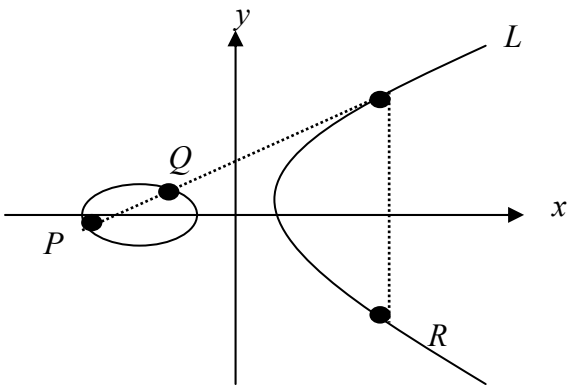

Fig 2.4 P and Q are two distinct points

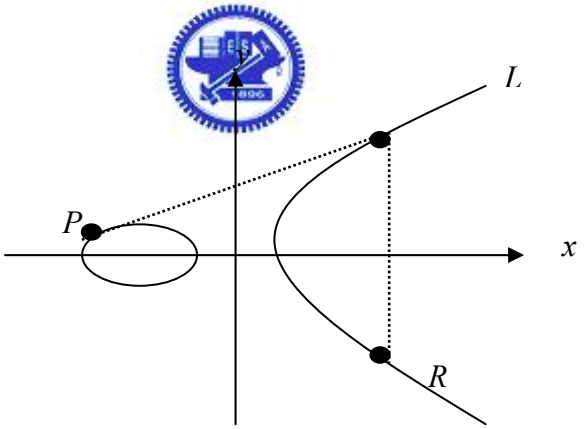

Fig 2.5 the addition of an elliptic curve point

First at all, we have to find the slope of $\overline{PQ}$, where $P \neq Q$ or the tangent line of P, where $P = Q$. We show as following:

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & \text{if } P \neq Q, \text{ where } \lambda \text{ is slope of line } \overline{PQ}. \\ \dfrac{3x_1^2 + a}{2y_1} & \text{if } P = Q, \text{ where } \lambda \text{ is tangent line of } P. \end{cases} \quad (2.3)$$

The equation of line L is $y = \lambda x + v$. The $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ is on L so that: $y_1 = \lambda x_1 + v$ and $y_2 = \lambda x_2 + v$. We substitute $y = \lambda x + v$ into the equation (2.1), getting the following:

$$(\lambda x + v)^2 = x^3 + ax + b$$

$$x^3 - \lambda^2 x^2 + (a - 2\lambda v)x + b - v^2 = 0 \qquad (2.4)$$

x_1 and x_2 are two roots of equation (2.4), which are real. As the result, the third root, said x_3, must also be real.

$$(x - x_1)(x - x_2)(x - x_3)$$

$$= x^3 - (x_1 + x_2 + x_3)x^2 + (x_1x_2 + x_2x_3 + x_1x_3)x - x_1x_2x_3 = 0 \qquad (2.5)$$

Comparing equation (2.4) and (2.5), we know that $\lambda^2 = x_1 + x_2 + x_3$. Hence,

$$x_3 = \lambda^2 - x_1 - x_2$$

The slope $\lambda = \dfrac{y_2 - y_1}{x_2 - x_1} = \dfrac{-y_3 - y_1}{x_3 - x_1}$; hence:

$$y_3 = \lambda(x_1 - x_3) - y_1$$

The rules for the sum of two points and the double of one point, we summarize as follows: for all $P, Q \in E(\mathbf{Z}_p^*)$ [Han04]:

1. $P + O = P$
2. If $P = (x, y)$, then the point $(x, -y)$ denoted as $-P$ and $P + (-P) = O$
3. Let $P = (x_1, y_1)$ and $Q = (x_2, y_2)$, where $P \neq Q$, then $P + Q = (x_3, y_3)$ where

$$\begin{cases} x_3 = \lambda^2 - x_1 - x_2, \\ y_3 = \lambda(x_1 - x_3) - y_1, \text{ where the slope } \lambda \text{ shows as equation (2.2).} \end{cases}$$

4. Let n be the smallest integer such that $nP = O$, then n is the order of P over E.

Elliptic curve cryptography (ECC) [Kob87] is a public-key cryptography based on the elliptic curve discrete logarithm problem (ECDLP) proposed by Neal Koblitz and Victor Miller in 1985. Given an elliptic curve E, over a Galois field GF(q), the operation "+" is defined as above paragraph and the operation "*" defined as $Z \times E(q) \rightarrow E(q)$ where $E(q)$ is rational points form (x, y), and both x and y are in GF(q). If P is some point in $E(q)$, then we define:

$$2*P = P + P,$$

$$3*P = 2*P + P = P + P + P, \text{ and so on.}$$

The ECDLP is then to determine integer k in $k*P = Q$, where P and Q are the given points. For a specific base point G is selected and published for use with the curve $E(q)$, Alice chooses a private key k as random integer and then the value $P = k*G$ is published as the public key. To encrypt a message m to Alice, we show as follows:

1. Convert message m (where $0 \le m < \frac{n}{2^k}$) to into an element P_m of $E(q)$
   - We append k bits at the end of the message
   - Compute $x = 2^k m + i$, for $i = 0, 1, \ldots,$ until $(\frac{x^3 + ax + b}{p}) = 1$.
2. Choose a random integer r
3. Calculate ciphertext $C_m = \{rG, P_m + rP\}$
4. Send the ciphertext C_m to Alice

Alice can decrypt ciphertext by multiplying the first point in the pair of Alice's secret key and subtracts the result from the second point:

$$P_m + rP - k(rG)$$

$$= P_m + r(kG) - k(rG)$$

$$= P_m.$$

2.4 One way hash functions

A one-way hash function, $h(m)$, operates on an arbitrary-length message m, and returns a fixed-length hash value, called digest. One-way hash functions are widely deployed in electronic mail, electronic funds transfer, software distribution, data storage, and other applications, which require the assurance of data integrity.

2.4.1 Secure Hash Standard

SHA, one kind of popular one-way hash functions, was originally applied to DSA (Digital Signature Algorithm), issued by the NIST and published as a federal information processing standard (FIPS PUB 180) in 1993; a revised version was issued as FIPS PUB 180-1 in 1995 [NIST95] and is generally referred to as SHA-160. SHA and SHA-160 operate on an arbitrary-length message as input; and then output a 160-bit digest.

FIPS 180-2 [NIST02] is announced by NIST on May 30, 2001. FIPS 180-2 is a strengthened version of the SHA-160, which offers four secure hash algorithms including SHA-160, SHA-224, SHA-256, SHA-384, and SHA-512. Table 2.4 presents the basic properties of FIPS 180-2.

Table 2.4 Comparison between all SHA-serial algorithms

Algorithm	Message Size	Block Size	Word size	Message Digest Size	Security*
SHA-160	$<2^{64}$	512	32	160	80
SHA-224	$<2^{64}$	512	32	224	112
SHA-256	$<2^{64}$	512	32	256	128
SHA-384	$<2^{128}$	1024	64	384	192
SHA-512	$<2^{128}$	1024	64	512	256

Note: *In this context, “security” refers to the fact that a birthday attack on a message digest of size n produces a collision with a work factor of approximately $2^{n/2}$.

The input of SHA and SHA-160 is processed in a 512-bit message block [NIST95]. First at all, for a 512-bit block, append padding and length after the message. The message block is transformed from 16 32-bit words (m_0, m_1, …, m_{15}) to 80 32-bit words (w_0, w_1, …, w_{79}) by the following algorithm: The difference between SHA and SHA-160 is that SHA-160 rotates 1 bit left: ROTL^1.

$$w_t = m_t \qquad ,0 \le t \le 15$$
$$w_t = \mathrm{ROTL}^1(w_{t-3} \oplus w_{t-8} \oplus w_{t-14} \oplus w_{t-16}) \quad ,16 \le t \le 79$$

Each of the 80 steps of the processing one 512-bit block form as:

$$A, B, C, D, E \leftarrow [\mathrm{ROTL}^5(A) + f_t(B,C,D) + E + w_t + k_t], A, \mathrm{ROTL}^{30}(B), C, D$$

Where f_t defines in belowing section and the logical operators (AND, OR, NOT, XOR) are represented by the symbols ($\wedge$, $\vee$, $\neg$, $\oplus$). k_t are constants, please refer to [NIST02].

$$f_t(x,y,z) = \mathrm{Ch}(x,y,z) = (x \wedge y) \vee (\neg x \wedge z) \qquad ,0 \le t \le 19$$
$$f_t(x,y,z) = \mathrm{Parity}(x,y,z) = x \oplus y \oplus z \qquad ,20 \le t \le 39$$
$$f_t(x,y,z) = \mathrm{Maj}(x,y,z) = (x \wedge y) \vee (x \wedge z) \vee (y \wedge z) \qquad ,40 \le t \le 59$$
$$f_t(x,y,z) = \mathrm{Parity}(x,y,z) = x \oplus y \oplus z \qquad ,60 \le t \le 79$$

The output of each round becomes initial value of next round until finish whole blocks. The final output is the concatenation of A, B, C, D, E.

2.4.2 Analyze SHA-160 in message schedule

We examine the changes from SHA to SHA-160 and discover the decay phenomenon with the application of a message schedule's judgment when inspecting how SHA-160 generates message schedule actually.

One reason from SHA to SHA-160: Firstly, we define notation $x^n = \mathrm{ROTL}^{n \bmod 32}(x)$. The message schedule of w_t of SHA-160 and SHA shall be prepared respectively as follows:

Table 2.5 Different message block between SHA and SHA-160

SHA	SHA-160
$w_t = m_t$,$0 \le t \le 15$	$w_t = m_t$,$0 \le t \le 15$
$w_t = w_{t-3} \oplus w_{t-8} \oplus w_{t-14} \oplus w_{t-16}$,$16 \le t \le 79$	$w_t = \mathrm{ROTL}^1(w_{t-3} \oplus w_{t-8} \oplus w_{t-14} \oplus w_{t-16})$,$16 \le t \le 79$

The other reason why ROTL^1 function can upgrade the security level is the increase of involved terms of m_t. For example, when comparing w_{27} in both SHA and SHA-160 (shown as follows), there are only 6 terms involved in SHA compared with 14 terms involved in SHA-160.

Table 2.6 w_{27} in SHA and in SHA-160

SHA involved 6 terms	$w_{27} = m_2 \oplus m_3 \oplus m_4 \oplus m_7 \oplus m_8 \oplus m_{15}$
SHA-160 involved 14 terms	$w_{27} = {m_2}^4 \oplus {m_3}^2 \oplus {m_4}^4 \oplus ({m_5}^2 \oplus {m_5}^3) \oplus {m_7}^3 \oplus {m_8}^2 \oplus ({m_{10}}^2 \oplus {m_{10}}^4) \oplus ({m_{11}}^1 \oplus {m_{11}}^2) \oplus ({m_{13}}^1 \oplus {m_{13}}^3) \oplus {m_{15}}^4$

W_{27} becomes independent of m_5 in the end even though m_5 has been involved twice in SHA. But in SHA-160, m_5 is involved under ROTL function thus ${m_5}^2$ and ${m_5}^3$ will not be eliminated. Belowing is a figure comparing the number of terms involved in message schedules of both SHA and SHA-160. *X*-axis presents the index, and *y*-axis presents the number of terms.