CHAPTER 5

EFFICIENT PROPRIETARY CERTIFICATION PROCESS AND SECURED THEFT-PROTECTED PROPRIETARY CERTIFICATES

5.1 INTRODUCTION

This chapter concerns the digital certificate based authentication scheme. This chapter presents the proprietary certification process using Elliptic Curve Discrete Log Problem. Also this chapter focuses the vulnerability in the existing *theft-protected digital certificates* and a new scheme is presented.

Digital certificates play a vital role in public key cryptography, and are commonly used in electronic commerce applications. The Certification Authority (CA) will issue the certificate to the client after successful completion of the secured protocol called “certification process”. This certificate will be generally installed in the user’s browser and will be used if the user wants to access a particular website. If the accessing website is a paid one then traditional certificate, however, are not secure against certificate lending. This type of abuse is a concern in several types of applications, such as those related to digital rights management (DRM). Proprietary certificates were introduced by Jakobsson et al (2002) to discourage sharing of access rights to subscription-based resources. A proprietary certificate is a certificate that contains some information related to another certificate called collateral certificate. This collateral certificate may contain the proprietors more

sensitive information which he/she doesn't want to reveal to any one. If Alice (proprietor) reveals the proprietary secret key to Bob (not having access rights) to access subscription-based resources then the corresponding collateral secret key of Alice will be automatically released to Bob which Alice doesn't like, therefore she discourages Bob by not giving her proprietary secret key.

While the original construction of proprietary certificates achieves its stated goal, but it overlooks the possible scenario in case of theft of the proprietary secret key that would lead to immediate loss of the collateral secret key also. Hence, theft of the proprietary key grants the intruder full access rights to all resources associated with both the proprietary and corresponding collateral keys. This approach punishes not only international sharing, but also accidental sharing. In Boldyreva and Jakobsson (2002) an additional property called theft protection was given as a solution. The core of their solution is based on the concept of time delay (T) for deriving the secret key so that the proprietor can do the necessary steps to change the secret key during this stipulated time T.

On the other hand it is also possible for Alice to give the access rights to Bob incidentally for time lesser than T as she knows that the collateral information will be revealed after the time T only. In this case the time lock concept may not be the proper solution and hence some other solution is required to solve this problem. In this paper we proposed a solution for the theft-protected proprietary certificate problem by introducing a mobile phone as an additional requirement for the proprietor and model the scheme with Verifiable Encryption as building blocks for our construction.

5.2 PROPRIETARY CERTIFICATION PROCESS

Let CA_1, CA_2 be the distinct certification authorities issuing the certificates for the proprietary and collateral services respectively. Let C_1, C_2 be the proprietary and collateral certificates of some user respectively which are publicly available. Let pk_1, sk_1 are the public and secret keys of the user to obtain the proprietary certificate C_1 from CA_1. Let pk_2, sk_2 are the public and secret keys of the user to obtain the collateral certificate C_2 from CA_2. A certification authority CA_1 wishes to issue a proprietary certificate C_1 to a certain user. The user has to provide a second certificate C_2, issued by CA_2 as collateral. This process is shown in Figure 5.1.

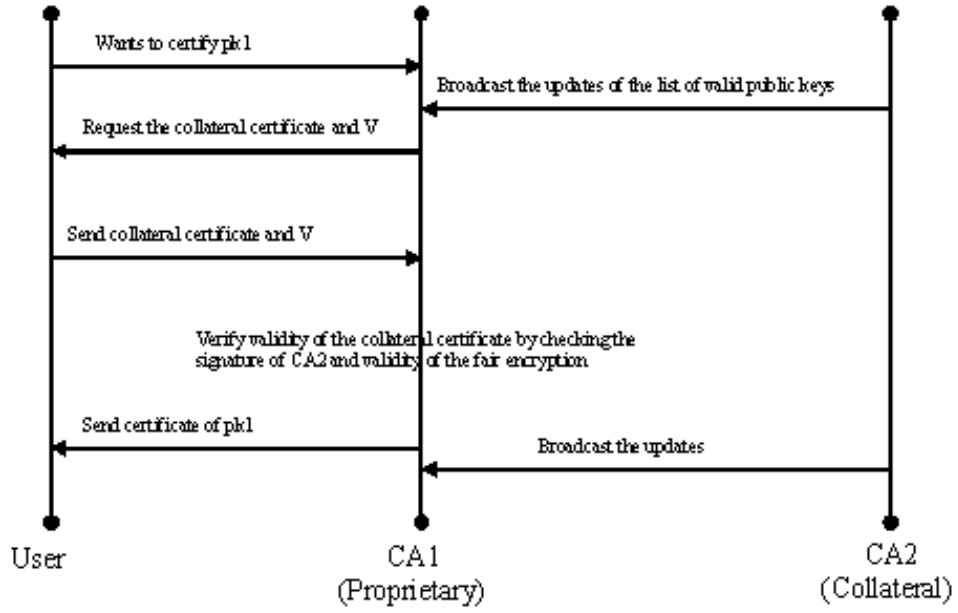

Figure 5.1 Overview of issuing proprietary certificates

The user sends the secret key sk_2 of the second certificate C_2 encrypted with the public key pk_1 to CA_1. Since sk_2 is encrypted with pk_1 to decrypt it requires sk_1 which is the secret key of the user, no information about

sk_2 will be revealed to CA_1. At the same time it is necessary for the user to convince the CA_1 about the correctness of the encrypted data.

The aim is to construct a proprietary certificate system that respects these requirements for heterogeneity and flexibility in public key infrastructure. Assume that certificate authorities publish directories containing public information on the certificates they have issued. In Jakobsson et al (2002) the desirable properties of the system are given, they are Non-transferability, Cryptosystem agility, Locality, Efficiency and Security. Also this showed that how to extend the regular certificate to make it proprietary one, namely being linked to the collateral certificate. It was suggested that verifiable encryption can be used for the implementation of proprietary certificates.

Let us assume that the user obtained the certificate for his collateral public key pk_2 from CA_2. In order to certify the public key pk_1, the certification authority CA_1 asks the user to present the certificate of another key pk_2 issued by CA_2 which he uses for some other services (to be considered as collateral) and the value $V = Ver_{pk_1}(sk_2)$ which is the verifiable encryption of his collateral secret key sk_2 under pk_1.

If CA_1 agrees to use this certificate as collateral, then verifies the validity of the collateral certificate by checking the signature of CA_2 and validity of the verifiable encryption. The properties of verifiable encryption ensure that CA_1 does not learn any information about the user's collateral secret key while being able to verify whether this cipher text is valid. CA_1 also needs to be sure that pk_2 is still a valid key. It is assumed that CA_2 broadcasts the updates to the list of valid public keys. Thus CA_1 needs to check that pk_2 is still in that list. No direct interaction between CA_1 and CA_2 is required. If the

verification is successful, then CA_1 includes V and the encryption of pk_2 under pk_1 in the certificate in addition to standard information such as the user's identity information and pk_1. If the user shares sk_1 with another party, then that party can decrypt V and obtain sk_2. The weakness of the above approach is the accidental exposure of a proprietary secret key due to theft or intrusion which would immediately lead to the loss of the collateral key. Therefore, direct use of proprietary certificates would be risky since it imposes additional insecurity on the collateral secret key.

5.3 ZERO-KNOWLEDGE PROOF SYSTEMS

There are many situations where it is necessary to "prove" one's identity. Typical scenario is to login to a computer, to get access to an account for electronic banking or to withdraw money from an automatic teller machine. Older methods use password or PINs to implement user identification. Though successfully used in certain environments, these methods also have weakness. For example, anyone to whom we must give our password to be verified has the ability to use that password and impersonate us.

Zero-knowledge identification schemes provide a new type of user identification. It is possible for us to authenticate our self without giving to the authenticator the ability to impersonate us.

5.3.1 Interactive Proof Systems

There are two participants in an interactive proof system, the prover and the verifier. Prover knows some fact (e.g. a secret key sk of a public key cryptosystem) which is called prover's secret. In an interactive proof of knowledge, Prover whishes to convince the verifier that he knows the prover's secret. Prover and verifier communicate with each other through a

communication channel. Prover and verifier alternately perform the following steps:

1. Receiving a message from the opposite party.
2. Performing some computation.
3. Sending a message to the opposite party.

Usually, prover starts and verifier finishes the protocol. In the first move, prover does not receive a message. The interactive proof may consist of several rounds. This means that the protocol specifies a sequence of moves, and this sequence is repeated a specified number of times. Typically, a move consists of a challenge by verifier and a response by prover. Verifier accepts or rejects verifier's proof, depending on whether prover successfully answers all of verifier's challenges.

5.3.2 Interactive Proof Systems Requirements

For the interactive proof systems, the following requirements are important:

1. Completeness: Both the prover and the verifier follow the protocol and the verifier always accepts the proof if it is true.
2. Soundness: As long as the verifier follows the protocol, the verifier will always reject the proof if the fact is false.
3. Zero knowledge: As long as the prover follows the protocol, the verifier learns nothing about the fact being proved, except that it is true, even if the verifier does not follow the protocol. In the zero knowledge proof, even if the exchange is recorded, the verifier cannot later prove the fact to anyone else. Biehl et al (1994) gave the tools for proving the zero knowledge protocol.

5.4 CHAUM AND PEDERSEN SCHEME

Let G_q denote the unique subgroup of Z_p^* of order q. The parameters p, q are primes such that q divides $p-1$, for instance $p = 2q+1$. Let $g, h \in G_q$ be publicly known bases. The prover selects a secret key $sk \bmod q$ and compute the cipher texts $c_1 = g^{sk}$, $c_2 = h^{sk}$ and send $c_1 \| c_2$ to the verifier. The prover must convince the verifier that the cipher text is obtained using the secret value sk but not any other value, and at the same time without revealing the value.

The protocol, described by Chaum and Pedersen (1992), is as follows:

1. The prover randomly chooses $t \in Z_q$ and sends (g^t, h^t) to the verifier
2. The verifier chooses a random challenge $c \in Z_q$ and sends it to the prover
3. The prover, then sends $s = t - c.sk \bmod q$ to the verifier
4. The verifier accepts the proof if $g^t = g^s c_1^{\,c}$ and $h^t = h^s c_2^{\,c}$

This is because

$$g^s c_1^{\,c} = g^{t-c.sk}.(g^{sk})^c = g^{t-c.sk+c.sk} = g^t \text{ and}$$
$$h^s c_2^{\,c} = h^{t-c.sk}.(h^{sk})^c = h^{t-c.sk+c.sk} = h^t$$

In step 3, the prover sends the secret value sk in a disguised manner. That is from the value $s = t - c.sk \bmod q$, sk cannot be found out. Hence without revealing the secret value, the prover can convince the verifier

using the above Zero Knowledge Proof (ZKP). This scheme is used to model the zero knowledge proof using the elliptic curve cryptosystem.

5.5 MATHEMATICAL MODEL

Let E: $y^2 = x^3 - 120x - 448$ is an Elliptic Curve cyclic group of order ρ, generated by B. Define $R = \{(w, A) \in W \times E : wB = A\}$, is the binary relation, where $W = [\rho]$. There are several advantages in using elliptic curves for cryptography. Elliptic curve cryptosystems with smaller key sizes appear to be just as secure as ``classical'' cryptosystems with much larger key sizes, so elliptic curve cryptosystems can be more efficient. Another advantage, is that elliptic curve cryptosystems appear to be vastly more secure over ``large finite fields of characteristic 2" than RSA, which is very important in practical applications. Some mobile phones also use elliptic curve cryptography.

The participating entities are the prover and the verifier. The agenda is that the prover encrypts secret information sk and sends to the verifier and subsequently convinces the verifier that the cipher text indeed decrypts to the secret value without compromising the secrecy. This protocol makes use of Elliptic curves cryptosystem for encrypting the message. That is the prover encrypts the secret information as

$$skB = A \tag{5.1}$$

and sends A to the verifier. Now the prover wishes to convince the verifier that the A will obtain using the repeated addition of the secret value sk, without revealing sk. This is achieved using the following protocol

1. The prover randomly chooses $m \in [-\rho, \rho]$ and compute

$$S = mB \tag{5.2}$$

2. The verifier chooses a random challenge c and sends to the prover.

3. The prover sends the response by computing
$$r = m - c.sk \quad (5.3)$$

4. The verifier accepts if
$$S = rB + cA \quad (5.4)$$

Equation (5.4) holds because, using (5.1), (5.2), (5.3) we get

$$rB + cA = (m - c.sk)B + c.(skB) = mB - c.sk.B + c.sk.B = mB = S$$

5.6 VERIFIABLE ENCRYPTION SCHEME BASED ON ECDLP

The verifiable encryption scheme based on ECDLP consists of four processes viz. key generation, encryption, decryption and the protocol. The key generation is explained in section 5.6.1, encryption and decryption scheme is explained in section 5.6.2 and the protocol is given in section 5.6.3.

5.6.1 Key Generation

Let $[a]$ be the set consisting of elements $\{0,1,...,a-1\}$, $H_{hk}(.)$ be the keyed hash function that uses a key hk, p^1, q^1 be the prime numbers, such that $p = 2p^1 + 1, q = 2q^1 + 1$ are primes. Let $n = pq$, $n^1 = p^1q^1$. The prover chooses the secret keys $x_1, x_2, x_3 \in [n^2/4]$ at random and select $g^1 \in Z^*_{n^2}$ and computes $g = (g^1)^{2n}$. The public keys are calculated using the formulae $y_1 = g^{x_1}$, $y_2 = g^{x_2}$, $y_3 = g^{x_3}$. Now $h = (1 + n \bmod n^2)$ is computed. The public keys are $y_1, y_2, y_3, n, g, hk)$ and the secret keys are x_1, x_2, x_3.

5.6.2 Encryption Scheme

Let $sk_2 \in [n]$ be the collateral secret key that the prover wants to convince the verifier. Let $L \in \{0,1\}^*$ is the Label. To encrypt the collateral secret key $sk_2 \in [n]$ with label $L \in \{0,1\}^*$, the prover choose a random $r \in [n/4]$ and computes

$$u = g^r \tag{5.5}$$

$$e = y_1^r h^{sk_2} \tag{5.6}$$

$$v = (y_2 y_3^{H_{hk}(u,e,L)})^r \tag{5.7}$$

The cipher text is (u,e,v).

To decrypt a cipher text (u,e,v) with label L, the verifier first checks that

$$u^{2(x_2 + H_{hk}(u,e,L)x_3)} = v^2 \tag{5.8}$$

If this does not hold then outputs “reject” and halt. Then the verifier calculates

$$t = 2^{-1} \bmod n \tag{5.9}$$

and computes

$$\tilde{m} = (e / u^{x_1})^{2t} \tag{5.10}$$

If $$\tilde{m} = h^{sk_2} \tag{5.11}$$

for some $sk_2 \in [n]$ then output sk_2; otherwise, output “reject”.

5.6.3 The Protocol

Let (N,G,H) be the augmented public key where $N = PQ$ is the product of two safe primes P,Q, such that $P = 2P^1 + 1$, $Q = 2Q^1 + 1$ and $N^1 = P^1Q^1$. Let $Y_{N^1} \subset Z^*_{N^2}$ is a subgroup of Z^*_N. Consider the elliptic curve $y^2 = x^3 - 120x - 448$. This curve forms a cyclic group E of order ρ. Let B be the generator of the elliptic curve point set E. Let $G, H \in Y_{N^1}$ are the two generators of Y^1_N. Let $W = [\rho]$ with $\rho < n$. Define the binary relation

$$R = \{(w, A) \in W \times E : wB = A\}$$

The common input of the prover and verifier is the public key $(y_1, y_2, y_3, n, g, hk)$, the augmented public key (N,G,H), a group element $A \in E$, a cipher text (u,e,v) and a label L.

The prover has additional inputs

$$sk_2 B = A \tag{5.12}$$

and

$r \in [n/4]$ such that (5.5), (5.6) and (5.7) are satisfied.

The prover chooses a random $s \in [n/4]$, $r^1 \in [-n, n]$, $s^1 \in [-N, N]$, $m^1 \in [-\rho, \rho]$ and computes

$$T = G^m H^s \tag{5.13}$$

$$u^1 = g^{2r^1} \tag{5.14}$$

$$e^1 = y_1^{2r^1} h^{2m^1} \tag{5.15}$$

$$A^1 = m^1 B \tag{5.16}$$

and $$T^1 = G^{m^1} H^{s^1} \tag{5.17}$$

The prover sends $(T, u^1, e^1, v^1, A^1, T^1)$ to the verifier.

The verifier chooses a random challenge c and sends to the prover.

The prover computes

$$\tilde{r} = r^1 - cr \tag{5.18}$$

$$\tilde{s} = s^1 - cs \tag{5.19}$$

$$\tilde{m} = m^1 - csk_2 \tag{5.20}$$

and sends $(\tilde{r}, \tilde{s}, \tilde{m})$ to the verifier.

The verifier checks whether the relations

$$u^1 = u^{2c} g^{2\tilde{r}} \tag{5.21}$$

$$e^1 = e^{2c} y_1^{2\tilde{r}} h^{2\tilde{m}} \tag{5.22}$$

$$v^1 = v^{2c} (y_2 y_3^{H_{hk}(u,e,L)})^{2\tilde{r}} \tag{5.23}$$

$$A^1 = \tilde{m} B + cA \tag{5.24}$$

$$T^1 = T^c G^{\tilde{m}} H^{\tilde{s}} \tag{5.25}$$

and $$-n/4 < \tilde{m} < n/4 \tag{5.26}$$

holds.

If any of them does not hold, the verifier stops and output 0. If all are verified the output is 1.

5.7 THEFT PROTECTED PROPRIETARY CERTIFICATES

To get the proprietary certificate, the user has to send its collateral secret key sk_2 to the Certification Authority. To achieve confidentiality the user encrypts its secret using the public encryption scheme and send the corresponding cipher text to the Certification Authority. Now the user has to convince the Certification Authority that indeed the cipher text decrypts to its original secret key sk_2. The process has been illustrated in the following Figure 5.2.

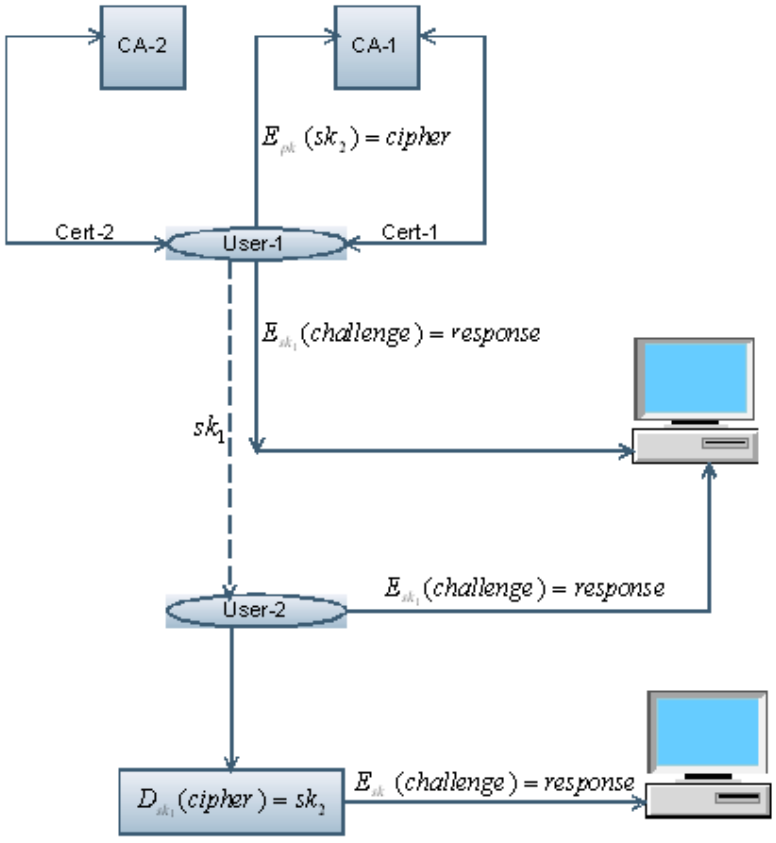

Figure 5.2 Verifiable encryption scheme for certification process

5.7.1 Making Proprietary Certificate Theft Protected

Min Wu's (2007) authentication protocol has been modified to get the solution for the accidental theft of proprietary secret key. Assume that the security proxy (P) is capable of generating a secured 'number generated once' (nonce) called as "challenge" and verify the dynamically generated secret random "challenge" concatenated with the originally stored secret key sk_2 for validating the authentication process. Illustration of the model is given in Figure 5.3. The process has seven steps:

1. The user (U) directs Internet Kiosk's browser (K) to contact the security proxy server (P).
2. U produces the certificate C_1 to K, which sends it to P.
3. P randomly chooses a "challenge" and sent it to user's mobile (M) as an SMS message.
4. U got the challenge from M
5. The user directs K's browser to contact P
6. U types the secret key sk_2 concatenated with the challenge which will be send to P.
7. Once authenticated, P operates like a traditional web proxy.

In this model, if Alice accidentally lost her proprietary secret key and if Charlie (not having access rights) got that secret key then after producing the public certificate C_1 to K, the system will wait for the new secret key = (sk_2||challenge) to enter and at the same time the corresponding challenge will be sent to Alice's mobile phone. From the unusual SMS message from the Proxy server, Alice might conclude that her secret key has been accidentally lost and someone is trying to impersonate her identity.

Hence she may take necessary steps to change that secret key; moreover Charlie doesn't gain any access form the security proxy as he couldn't know the "challenge".

If Alice incidentally wants to give her secret key and mobile phone to Bob, then he can get the corresponding collateral secret key of Alice as discussed in Figure 5.3, which Alice doesn't like, therefore she discourages Bob by not giving her proprietary secret key and the mobile phone.

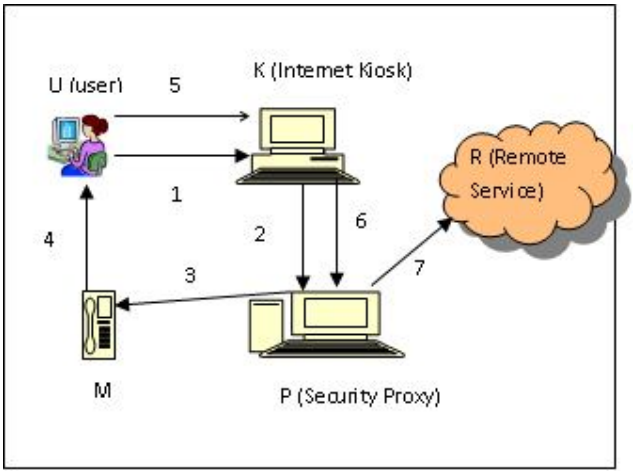

Figure 5.3 A new approach to avoid penalizing accidental sharing

The process had been analyzed for the two most likely security threats: (1) K remembers connection information for replay attack at a later time; (2) P receives two simultaneous connections from different kiosks, each claiming to be the same user. This can be avoided by using a unique session name (SN) for each user's session, and a nonce (N) that is transmitted to M with the SMS message. N prevents forged replies from an attacker who knows SN but does not have possession of M. Security of the system depends on the security of messages sent by SMS. The security of this system also depends upon the fact that U is in possession of M. It is a reasonable

assumption: when people lose their mobile phones, they are typically reported lost and deactivated. Once deactivated, M will no longer be able to receive SMS messages destined for U.

5.8 COMPARISION BEWEEN ECDLP AND DLP

Given a pair of points (P, mP) find the integer m, is the problem with a very different nature from that of point multiplication. This problem is called Elliptic-Curve Discrete Logarithm Problem. It is widely believed that the ECDLP is difficult to solve when the point P has a large prime order. For a curve defined over F_q it is very easy to devise a large prime r of size slightly less than q such that $E(F_q)$ contains a subgroup of order r. In the case of Discrete Logarithm Problem (DLP) in a finite field, there exist algorithms called index calculus for solving the problem. The time complexity of an index calculus method for discrete logarithm in a finite field F_q has a sub-exponential expression $\exp\left(\frac{\log q}{2}\right)$.

In specific, if an elliptic curve over a finite field F_q with $q \approx 2^{160}$, the difficulty of the ECDLP will be expressed by a 2^{80} value. To obtain a similar difficulty for the DLP in a finite field, the sub-exponential expression will reduce to $q \approx 2^{1000}$.Hence if we use the ECDLP relation, the key size may be reduced to 6.25 time that of the key size used for DLP.

5.9 NUMERICAL EXAMPLE

Consider the elliptic curve equation $y^2 = x^3 - 10x - 8 (\text{mod} 11)$, the elliptic curve point set E consist of the following 18 points. The set of points in the elliptic curve $y^2 = x^3 - 10x - 8 (\text{mod} 11)$ is given in Table 5.1.

Table 5.1 Elliptic curve points set elements for

$$y^2 = x^3 - 10x - 8 (\text{mod} 11)$$

(0, 5)	(0, 6)	(1, 4)	(1, 7)	(3, 0)	(4, 4)
(4, 7)	(5, 1)	(5, 10)	(6, 4)	(6, 7)	(7, 1)
(7, 10)	(9, 2)	(9, 9)	(10, 1)	(10, 10)	*O*

Since B generates all the points in the set E as shown in Table 5.2, $B = (10,10)$ be the generator of order $\rho = 18$.

Table 5.2 Repeated doubling of the point B = (10,10)

1B = (10, 10)	2B = (6, 4)	3B = (0, 5)	4B = (4, 4)	5B = (9, 2)
6B = (1, 7)	7B = (5, 10)	8B = (7, 1)	9B = (3, 0)	10B = (7,10)
11B = (5, 1)	12B = (1, 4)	13B = (9,9)	14B = (4,7)	15B = (0,6)
16B = (6, 7)	17B = (10,1)	18B = *O*		

The numerical values of the encryption process are shown in Table 5.3.

Table 5.3 Numerical values for the variables in encryption scheme

$p^1 = 3$	$q^1 = 5$	$p = 7$	$q = 11$	$n = 77$
$n^1 = 15$	$n^2 = 5929$	$[n^2 / 4] = [1482]$	$x_1 = 1$	$x_2 = 2$
$x_3 = 3$	$g^1 = 2$	$g = 5714$	$y_1 = 5714$	$y_2 = 4722$
$y_3 = 4558$	$h = 78$	$sk_2 = 4$	$B = (10,10)$	$A = (4,4)$
$r = 1$	$L = 1$	$H_{hk}^{(u,e,L)} = 1$	$u = 5714$	$e = 4713$
$v = 606$	$v^2 = 5567$	$u^{2(x_2 + H_{hk}(u,e,L)x_3)} = 5567$	$t = 39$	$\tilde{m} = 309$
$h^{sk_2} = 309$				

The output is accepted since $h^{sk_2} = 78^4 \bmod 5929 = 309 = \tilde{m}$.

In the protocol the values used are shown in the Table 5.4.

$Z_{35}^* = \{1,2,3,4,6,8,9,11,12,13,16,17,18,19,22,24,26,27,29,31,32,33,34\}$

Let $Y_6 \subset Z_{35}^*$ be the subgroup and $Y_6 = \{1,11,16,19,24,34\}$

Let $G = 19$, $H = 24$ be the two generators of Y_6

Table 5.4 Numerical values for the variables in the protocol

$\rho = 18$	$W = [18]$	$P = 5$	$Q = 7$	$N = 35$
$N^1 = 6$	$s = 1$	$r^1 = 1$	$s^1 = 1$	$m^1 = 3$
$T = 34$	$u^1 = 4722$	$e^1 = 4414$	$v^1 = 5567$	$A^1 = (0,5)$
$T^1 = 11$	$c = 1$	$\tilde{r} = 0$	$\tilde{s} = 0$	$\tilde{m} = -1$
$u^{2c} g^{2\tilde{r}} = 4722$	$e^{2c} y_1^{2\tilde{r}} h^{2\tilde{m}} = 4414$	$v^{2c}(y_2 y_3^{H_{hk}(u,e,L)})^{2\tilde{r}} = 5567$	$A^1 = (0,5)$	$\tilde{m}B + cA = (0,5)$
$T^c G^{\tilde{m}} H^{\tilde{s}} = 11$	$-n/4 < -1 < n/4$	$G = 19$	$H = 24$	

The verifier is convinced for his challenge $c = 1$, since $\tilde{m}B + cA = -1(10,10) + 1(4,4) = (10,-10) + (4,4) = (10,1) + (4,4) = (0,5)$.

5.10 PERFORMACE EVALUATION

The performance of the scheme in terms of number of keys, computational complexity has been analyzed in this section. The following notation were used to analyze the performance of the scheme

- **E** is the time for modular Exponentiation
- **M** is the time for modular multiplication
- **I** is the time for a modular inverse Computation
- **H** is the time for performing a one-way Hash

Here the time for performing modular addition/subtraction computation is ignored. The performance evaluation of the proposed scheme and the programming output data are shown in Tables 5.5 and 5.6.

Table 5.5 Performance evaluation of the proposed scheme

Key Generation	4E+H
Encryption	5E+2M+H
Decryption	4E+2I+H
Prover Verification	10E+7M+H
Verifier Verification	13E+8M+H

Table 5.6 Comparison of time needed

Description	**Key Size**	**ECDLP (nano seconds)**	**DLP (nano seconds)**
Key Generation	4 bit	385803	397893
	8 bit	410887	420987
	16 bit	532205	582307
	32 bit	704304	708323
Encryption	4 bit	306256	313213
	8 bit	342376	372378
	16 bit	515779	525678
	32 bit	836358	856543
Prover's Computation	4 bit	625802	645809
	8 bit	673993	693543
	16 bit	721477	751976
	32 bit	902357	908754
Verifier's Computation	4 bit	633851	673482
	8 bit	703854	709856
	16 bit	758772	778675
	32 bit	935341	945342

5.11 INFERENCE AND CONCLUSION

Based on the numerical data, the time needed for the four major computation processes in this scheme viz. key generation, encryption process, prover's computation and verifiers computation had been plotted in Figures 5.4, 5.5, 5.6 and 5.7 respectively. ECDLP based scheme is compared with DLP scheme.

- It is inferred that the time needed for ECDLP based scheme is very similar to DLP based scheme; therefore no additional time is required for processing the proposed scheme.

- Implementation of this scheme using ECDLP will provide the same level of security with the key size which is reduced to 6.25time that of the key size used for DLP.

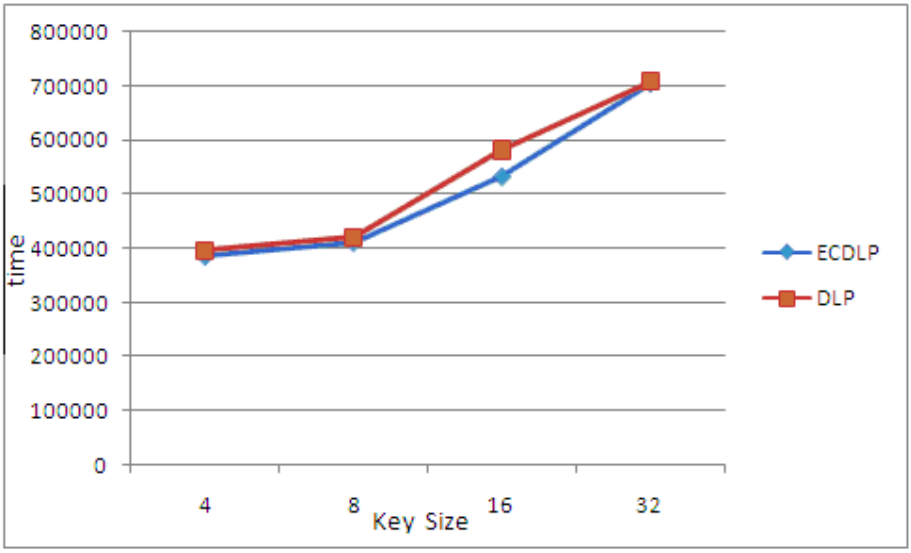

Figure 5.4 Key generation process

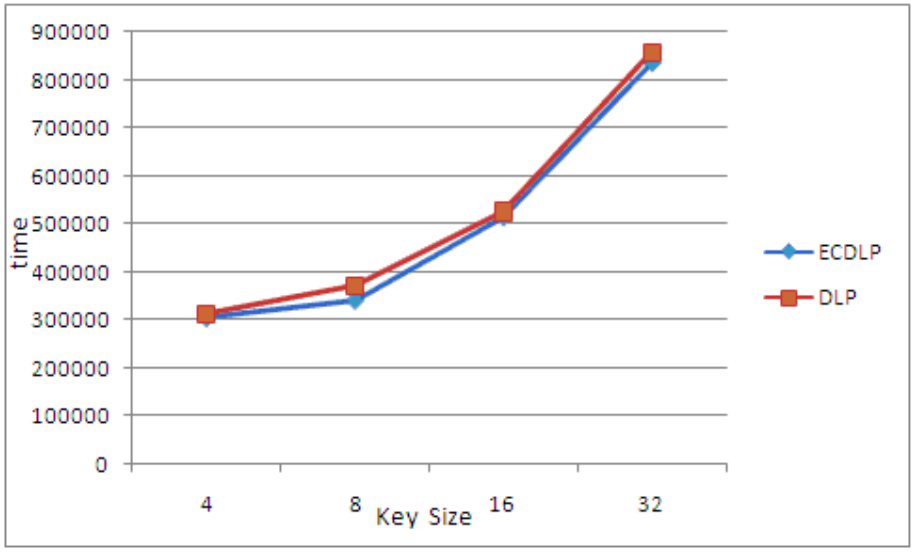

Figure 5.5 Encryption process

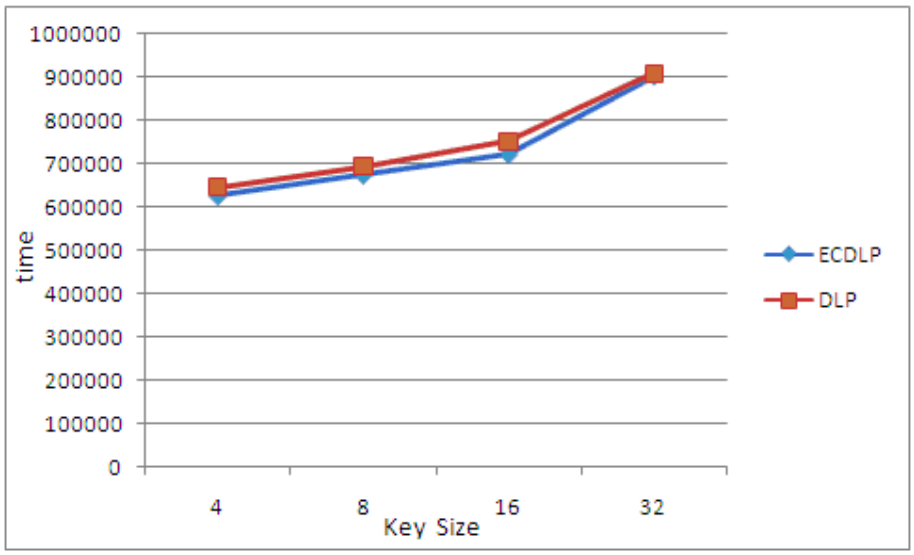

Figure 5.6 Prover's computation

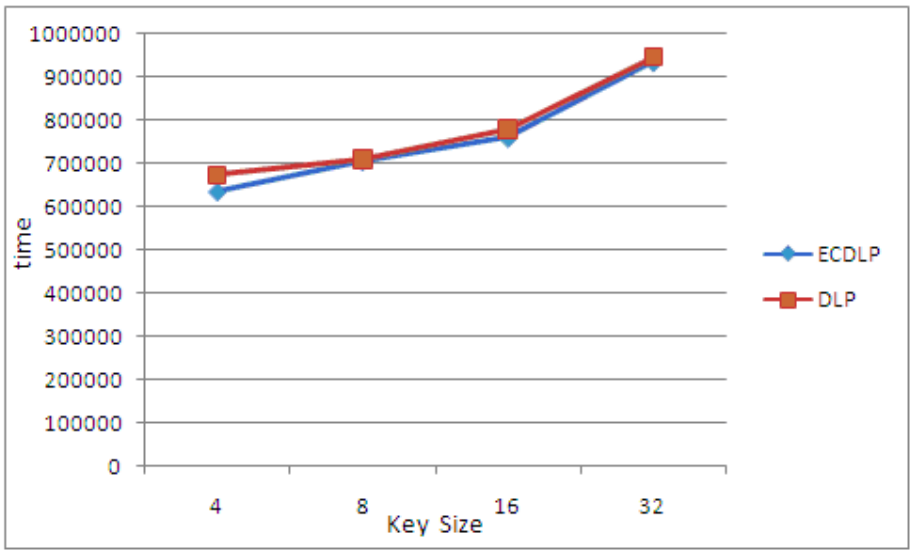

Figure 5.7 Verifier's computation