Chapter 2

Literature Survey

This chapter describes the popular cryptosystems and the cryptanalysis on these systems in detail. Further the research contributions in devising, developing or enhancing the algorithms in solving the DLP for the last 20 years are discussed.

The security goals on basic communication model are confidentiality, data integrity, data authentication, entity authentication and non repudiation. The cryptographic systems are basically designed to achieve these goals. Cryptology is the combination of cryptography and cryptanalysis. Cryptography deals with the design and analysis of mathematical techniques that enable secure communication in the presence of malicious adversaries while cryptanalysis deals with the secure systems (cryptosystems) for vulnerabilities. The cryptosystems can be divided into two types, such as symmetric key and asymmetric key (public key cryptosystems).

2.1 Public key cryptography

Public-key refers to a cryptographic mechanism. It has been named public-key to differentiate it from the traditional and more intuitive cryptographic mechanism known as: symmetric-key, shared secret, secret-key and also called private-key. Symmetric-key cryptography is a mechanism by which the same key is used for both encrypting and decrypting; it is more intuitive because of its similarity with the same key. This characteristic requires sophisticated mechanisms to securely distribute the secret-key to both parties. Differential [21, 55, 11, 12, 13] and linear
analysis [57, 58, 59] are the best known analysis on symmetric algorithms. Public-key on the other hand, introduces another concept involving key pairs: one for encrypting, the other for decrypting. This concept, is very clever and attractive, and provides a great deal of advantages over symmetric-key, such as simplified key distribution, digital signature, long-term encryption. Public-key is commonly used to identify a cryptographic method that uses an asymmetric-key pair: a public-key and a private-key. Public-key encryption uses that key pair for encryption and decryption. The public-key is made public and is distributed widely and freely. The private-key is never distributed and must be kept secret. Given a key pair, data encrypted with the public-key can only be decrypted with its private-key; conversely, data encrypted with the private-key can only be decrypted with its public-key. All public key cryptosystems are based on NP-problems such as DLP (Discrete Logarithm Problem), IFP (Integer Factorization Problem) to name a few. For example, the popular RSA public key cryptography depends on the IFP, while the Diffie-Hellman key exchange, Elliptic curve cryptosystem, El-Gamal and XTR are based on the DLP.

The popular cryptosystem RSA was publicly described in 1977 by Ron Rivest, Adi Shamir, and Leonard Adleman at MIT; the letters RSA are the initials of their surnames, listed in the same order as on the paper [74]. The Diffie-Hellman key agreement was invented in 1976 during a collaboration between Whitfield Diffie and Martin Hellman and was the first practical method for establishing a shared secret over an unprotected communications channel. Another popular public key cryptosystem is El-Gamal. It was described by Taher El-Gamal in 1984 [30].

RSA is the first algorithm known to be suitable for signing as well as encryption, and one of the first great advances in public key cryptography. RSA is widely used in electronic commerce protocols, and is believed to be secure given sufficiently long keys and the use of up-to-date implementations [63, 74, 26]. The security of the RSA cryptosystem is based on two mathematical problems: the problem of factoring large numbers (IFP) and the RSA problem. Full decryption of an RSA ciphertext is thought to be infeasible on the assumption that both of these problems are hard, i.e., no efficient algorithm exists for solving them. Providing security against partial decryption may require the addition of a secure padding
scheme. The RSA problem is defined as the task of taking \( e^{th} \) roots modulo a composite \( n \): recovering a value \( m \) such that \( c = m^e \mod n \), where \((n, e)\) is an RSA public key and \( c \) is an RSA ciphertext. Currently the most promising approach to solving the RSA problem is to factor the modulus \( n \). With the ability to recover prime factors, an attacker can compute the secret exponent \( d \) from a public key \((n, e)\), then decrypt \( c \) using the standard procedure. To accomplish this, an attacker factors \( n \) into \( p \) and \( q \), and computes \((p - 1)(q - 1)\) which allows the determination of \( d \) from \( e \). No polynomial-time method for factoring large integers on a classical computer has yet been found, but it has not been proven that none exists.

As of 2005, the largest number factored by a general-purpose factoring algorithm was 663 bits long, using a state-of-the-art distributed implementation. RSA keys are typically 1024 to 2048 bits long. Therefore, it is generally presumed that RSA is secure if \( n \) is sufficiently large. If \( n \) is 256 bits or shorter, it can be factored in a few hours on a personal computer, using software already freely available. Keys of 512 bits (or less) have been shown to be practically breakable in 1999 when RSA-155 was factored by using several hundred computers. A theoretical hardware device named TWIRL and described by Shamir and Tromer in 2003 called into question the security of 1024 bit keys. It is currently recommended that \( n \) be at least 2048 bits long.

On the other hand, the Diffie-Hellman, El-Gamal and XTR are based on DLP. The key exchange protocol is one of the most elegant ways of establishing secure communication between pair of users by using a session key. The session key, which is exchanged between two users assures the secure communication for later sessions. The first practical key exchange protocol is proposed by Diffie-Hellman. Since the introduction of key exchange protocol by Diffie-Hellman, various versions and improvements in key exchange protocol have been developed. El-Gamal is the popular and first cryptosystem based on DLP for both encryption and digital signatures. The XTR is another public key cryptosystem based on sub group DLP. The DLP defined on a sub group and the efficient representation of sub group elements assure the security of this system. Similarly, the elliptic curve cryptosystem is a popular public key cryptosystem based on ECDLP. The ECDLP
is a DLP defined on an elliptic group.

As of now, the DLP on integer field of 120 digits and $GF(2^n)$ of 607 bits are solved. This leads to choose 1024 bits long for Diffie-Hellman and El-Gamal keys. Since the XTR uses an efficient representation of subgroup elements, the keys are 170 bits of subgroup on 1024 bits of field. The ECC keys are much shorter and comparable with XTR with 160 to 200 bits. The ECDLP on the field of size 108 bits is solved so far.

### 2.2 DLP based public key cryptosystems

As discussed in the previous section the DLP is the basis for many public key cryptosystems. The generalized DLP is defined as follows: For a given prime number $p$, a generator $g \in Z_p^*$ and an element $y \in Z_p^*$, the problem of finding an $x$ ($0 \leq x \leq p - 2$) such that $g^x \equiv y \mod p$ is known as the discrete logarithm problem. DLP is also defined over other groups such as the multiplicative group of Galois field $GF(p^n)$ and the collection of points defined by an elliptic curve over a finite field. For example the XTR cryptosystem is based on the DLP defined over a prime order sub group of $GF(p^6)$. The XTR uses the trace over $GF(p^2)$ to represents elements of subgroup of $GF(p^6)^*$ of the order $p^2 - p + 1$. The trace representation leads to achieve an efficient arithmetic on $GF(p^2)$. Even though the arithmetic are to be performed on $GF(p^2)$, the security is achieved on $GF(p^6)$ [53, 54]. Another example is the elliptic curve cryptosystems. They are based on the DLP defined over elliptic curve group. The above two methods uses the basic principle for generating the keys. Later they uses either Diffie-Hellman or El-Gamal for encryption.

#### 2.2.1 Diffie-Hellman

Diffie-Hellman key exchange (D-H) is a cryptographic protocol that allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel. This key can then be used to encrypt subsequent communications using a symmetric key cipher. Synonyms
of Diffie-Hellman key exchange include:

- Diffie-Hellman key agreement
- Diffie-Hellman key establishment
- Diffie-Hellman key negotiation
- Exponential key exchange

The scheme was first published publicly by Whitfield Diffie and Martin Hellman in 1976, although it later emerged that it had been separately invented a few years earlier within GCHQ, the British signals intelligence agency, by Malcolm J. Williamson but was kept classified. In 2002, Hellman suggested the algorithm be called Diffie-Hellman-Merkle key exchange in recognition of Ralph Merkle’s contribution to the invention of public-key cryptography (Hellman, 2002). Although Diffie-Hellman key agreement itself is an anonymous (non-authenticated) key-agreement protocol, it provides the basis for a variety of authenticated protocols, and is used to provide perfect forward secrecy in Transport Layer Security’s ephemeral modes.

**History**

Diffie-Hellman key agreement was invented in 1976 during a collaboration between Whitfield Diffie and Martin Hellman and was the first practical method for establishing a shared secret over an unprotected communications channel. Ralph Merkle’s work on public key distribution was an influence. John Gill suggested application of the discrete logarithm problem. It had been discovered by Malcolm Williamson of GCHQ in the UK some years previously, but GCHQ chose not to make it public until 1997, by which time it had no influence on research in academia. The method was followed shortly afterwards by RSA, another implementation of public key cryptography using asymmetric algorithms.

**Diffie-Hellman key exchange**

The key is assumed to be exchanged between two communicators, say Alice and Bob. Alice and Bob wishes to agree on a secret random element in the group
which could be used as a key for a higher speed symmetric algorithm like the
DES. They wish to make this agreement over an insecure channel without having
exchanged any information previously.

The public items are the group $G$, and an element $g \in G$ of large known order.
Based on this assumption the following procedure is carried out.

Alice generates a random integer $x_A \in 1 \ldots \#G - 1$ and sends to Bob the element
$g^{x_A}$
Bob generates a random integer $x_B \in 1 \ldots \#G - 1$ and sends the element $g^{x_B}$ to
Alice

Then Alice can compute
\[
g^{x_A x_B} = (g^{x_B})^{x_A}
\]
and Bob can compute
\[
g^{x_A x_B} = (g^{x_A})^{x_B}
\]

Thus, the secret key exchanged is $g^{x_A x_B}$ [29].

**Security**

The protocol is considered secure against eavesdroppers if the group $G$ and the
generator $g$ are chosen properly. This is currently considered difficult. An efficient
algorithm to solve the discrete logarithm problem would make it easy to compute
$a$ or $b$ and solve the Diffie-Hellman problem, making this and many other public
key cryptosystems insecure. The order of $G$ should be prime or have a large prime
factor to prevent the attacks based on the order of group. For this reason, a
Sophie Germain prime $q$ is sometimes used to calculate $p = 2q + 1$, called a safe
prime, since the order of $G$ is then only divisible by 2 and $q$. $g$ is then sometimes
chosen to generate the order $q$ subgroup of $G$, rather than $G$, so that the Legendre
symbol of $g^a$ never reveals the low order bit of $a$. If Alice and Bob use random
number generators whose outputs are not completely random and can be predicted
to some extent, then Eve’s task is much easier. The secret integers $a$ and $b$ are
discarded at the end of the session. Therefore, Diffie-Hellman key exchange by
itself trivially achieves perfect forward secrecy because no long-term private keying
material exists to be disclosed [29, 37, 28, 86, 36]. Perfect Forward Secrecy (or PFS) is the property that ensures that a session key derived from a set of long-term public and private keys will not be compromised if one of the (long-term) private keys is compromised in the future. Forward secrecy has been used as a synonym for perfect forward secrecy.

### 2.2.2 El-Gamal cryptosystem

The El-Gamal encryption system is an asymmetric key encryption algorithm for public-key cryptography. It was described by Taher El-Gamal in 1984 [30]. El-Gamal encryption is used in the free GNU Privacy Guard software, recent versions of PGP, and other cryptosystems. The Digital Signature Algorithm is a variant of the El-Gamal signature scheme. El-Gamal encryption can be defined over any cyclic group $G$. Its security depends upon the difficulty of a certain problem in $G$ related to computing discrete logarithms.

**El-Gamal cryptosystem**

Alice wishes to send a message, say $m$, to Bob. The message $m$ is assumed to be encoded as an element in the group. Bob has a public key consists of $g$ and $h = g^x$ where $x$ is the private key of Bob.

Alice generates a random integer $k \in 1...\#G − 1$ and computes

$$a = g^k, b = h^k m$$

Alice sends the cipher text $(a, b)$ to Bob.

Bob can receive the message from the equation as follows

$$ba^{-x} = h^k mg^{-kx} = g^{xk−zk} m = m$$

**Security**

Analysis based on the best available algorithms for both factoring and discrete logarithms show that the RSA system and the El-Gamal system have similar se-
curity for equivalent key lengths. The security of the El-Gamal scheme depends on the properties of the underlying group $G$ as well as any padding scheme used on the messages. If the Computational Diffie-Hellman assumption holds the underlying cyclic group $G$, then the encryption function is one-way. If the Decisional Diffie-Hellman assumption (DDH) holds in $G$, then El-Gamal achieves semantic security. Semantic security is not implied by the Computational Diffie-Hellman assumption (CDH) alone. The CDH assumption states that, given $(g, g^a, g^b)$ for a randomly-chosen generator $g$ and random $a, b \in \{0, \ldots, q - 1\}$, it is computationally intractable to compute the value $g^{ab}$. The DDH assumption states that, given $g^a$ and $g^b$ for randomly-chosen $a, b \in \mathbb{Z}_q$, the value $g^{ab}$ "looks like" a random element in the group $G$. This intuitive notion is formally stated by saying that the following two probability distributions are computationally indistinguishable (in the security parameter $q$): $(g^a, g^b, g^{ab})$, where $a$ and $b$ are randomly and independently chosen from $\mathbb{Z}_q$. $(g^a, g^b, g^c)$, where $a, b, c$ are randomly and independently chosen from $\mathbb{Z}_q$.

El-Gamal encryption is unconditionally malleable, and therefore is not secure under chosen ciphertext attack. For example, given an encryption $(c_1, c_2)$ of some (possibly unknown) message $m$, one can easily construct a valid encryption $(c_1, 2c_2)$ of the message $2m$. To achieve chosen-ciphertext security [27, 6, 7], the scheme must be further modified, or an appropriate padding scheme must be used. Depending on the modification, the DDH assumption may or may not be necessary [15].

Other schemes related to El-Gamal which achieve security against chosen ciphertext attacks have also been proposed. The Cramer-Shoup system is secure under chosen ciphertext attack assuming DDH holds for $G$. Its proof does not use the random oracle model. Another proposed scheme is DHAES [1], whose proof requires an assumption that is weaker than the DDH assumption.

**Digital signature schemes**

Digital signature schemes can be devised for data authentication, data integrity and to facilitate the provision of non-repudiation services. An entity $A$ would use
the signature generation algorithm, say \text{SIGN}, of a digital signature scheme and his private key \(d_A\) to compute the signature of a message as \(S = \text{SIGN}_{d_A}(m)\). Upon receiving \(m\) and \(S\) an entity \(B\) who has an authentic copy of \(A\)'s public key \(e_A\) uses a signature verification algorithm to confirm that \(S\) was indeed generated from \(m\) and \(d_A\). All public key cryptosystems provide elegant solution to key distribution, key management and the provision of non-repudiation. The popular digital signature schemes are DSA and DSS. The following paragraph discusses the El-Gamal digital signature.

El-Gamal digital signature:-

Bob wants to sign a message \(m\). He can use the public and private key pair \(h\) and \(x\) as discussed in the previous section. First he generates a random integer \(k \in 1,...,\#G - 1\) and computes

\[
a = g^k
\]

Then he computes a solution \(b\) to the congruence

\[
m \equiv xf(a) + bk \pmod{G}
\]

and sends the signature \((a, b)\) and message \(m\) to Alice.

Alice verifies the signature by checking the following equation holds [30].

\[
h \cdot f(a)^b = g^{xf(a) - kb} = g^m
\]

2.2.3 XTR cryptosystem

XTR stands for ECSTR. This is an abbreviation for Efficient and Compact Subgroup Trace Representation. It is a novel method that makes use of traces to represent and calculate powers of elements of a subgroup of a finite field. The XTR uses the trace over \(GF(p^2)\) to represents elements of the order \(p^2 - p + 1\) subgroup of \(GF(p^6)^*\), thereby achieving a 3 factor reduction. Also, the resulting calculations are appreciably faster than the standard representation.

Many cryptographic protocols used to be based on generator of a full multiplicative group of finite field. Schnorr introduced the idea of replace this generator by the generator of a relatively small subgroup of sufficiently large prime order \(q\).
The same idea is used in XTR. XTR uses a subgroup of prime order $q$ of the order $p^2 + p - 1$ subgroup of $GF(p^6)^*$. The latter group is referred as XTR supergroup and the subgroup of order $q$ as XTR subgroup.

**Selection of $p$ and $q$**

The primes $p$ and $q$ have to be selected in such a way that $q$ divides $p^2 - p + 1$ and such that the resulting fields and subgroups with stand known attacks. Furthermore in order to able to use the fast $GF(p^2)$ arithmetic, the $p$ should be $2$ mod $3$.

**XTR-Diffie-Hellman**

Let $p$, $q$ and $Tr(g)$ be shared XTR data, where $Tr(g)$ is a trace of generator $g$. If Alice and Bob want to agree upon a secrete key $K$ they do the following:

- Alice selects a random integer $a \in [2, q - 3]$ and compute the following equation with $n = a$ and $c = Tr(g)$
  \[ S_a(Tr(g)) = (Tr(g^{a-1}), Tr(g^a), Tr(g^{a+1}) \]
  and sends $Tr(g^a)$ to Bob.

- Bob selects a random integer $b \in [2, q-3]$ and compute the following equation with $n = b$ and $c = Tr(g)$
  \[ S_b(Tr(g)) = (Tr(g^{b-1}), Tr(g^b), Tr(g^{b+1}) \]
  and sends $Tr(g^b)$ to Alice.

- Alice compute the following with $n = a$ and $c = Tr(g^b)$
  \[ S_a(Tr(g^b)) = (Tr(g^{(a-1)b}), Tr(g^{ab}), Tr(g^{(a+1)b}) \]
  and determines the key $K$ as $Tr(g^{ab})$.

- Bob compute the following with to $n = b$ and $c = Tr(g^a)$
  \[ S_b(Tr(g^a)) = (Tr(g^{(a(b-1))}), Tr(g^{ab}), Tr(g^{a(b+1)}) \]
and determines the key $K$ as $Tr(g^{ab})$.

note: Refer [53, 54] for the computation of $S_n(c)$.

In the similar way XTR-El-Gamal also can be developed using the XTR parameters [53, 54].

**Security**

The best attacks are Pollards rho method in the order $q$ subgroup, or the Discrete Logarithm variant of the Number Field Sieve, another variant of ICM in the full multiplicative group $GF(p^6)^*$. With primes $p$ and $q$ of about $1024/6 \approx 170$ bits the security of XTR is equivalent to traditional subgroup systems using 170-bit subgroups and 1024-bit finite fields. But with XTR, subgroup elements can be represented using only about $2 \times 170$ bits, which is substantially less than the 1024-bits required for their traditional representation.

Full exponentiation in XTR is faster than full scalar multiplication in an Elliptic Curve Cryptosystem (ECC) over a 170-bit prime field, and thus substantially faster than full exponentiation in either RSA or traditional subgroup discrete logarithm systems of equivalent security. XTR keys are much smaller than RSA keys of comparable security. ECC keys allow a smaller representation than XTR keys, but in many circumstances (e.g. storage) ECC and XTR key sizes are comparable. Key selection for XTR is very fast compared to RSA, and orders of magnitude easier and faster than for ECC. As a result XTR may be regarded as the best of two worlds, RSA and ECC. It is an excellent alternative to either RSA or ECC in applications such as SSL/TLS (Secure Sockets Layer, Transport Layer Security), public key smartcards, WAP/WTLS (Wireless Application Protocol, Wireless Transport Layer Security), IPSEC/IKE (Internet Protocol Security, Internet Key Exchange), and SET (Secure Electronic Transaction).

The XTR key selection is very easy. This makes it easily feasible for all users of XTR to have public keys that are not shared with others, unlike ECC where a large part of the public key is often shared between all users of the system. Also, compared to ECC, the mathematics underlying XTR is straightforward, thus avoiding two common ECC-pitfalls: ascertaining that unfortunate parameter choices are
avoided that happen to render the system less secure, and keeping abreast of, and incorporating additional checks published in, newly obtained results. As a consequence the draft IKE protocol (part of IPSec) for ECC was revised.

XTR is the first method that uses $GF(p^2)$ arithmetic to achieve $GF(p^6)$ security, without requiring explicit construction of $GF(p^6)$. Let $g$ be an element of order $q > 6$ dividing $p^2 - p + 1$. Because $p^2 - p + 1$ divides the order $p^6 - 1$ of $GF(p^6)^*$ this $g$ generates an order $q$ subgroup of $GF(p^6)^*$. Since $q$ does not divide any $p^s - 1$ for $s = 1, 2, 3$, the subgroup generated by $g$ cannot be embedded in the multiplicative group of any true subfield of $GF(p^6)$. However the arbitrary powers of $g$ can be represented using a single element of the subfield $GF(p^2)$, and that such powers can be computed efficiently using arithmetic operations in $GF(p^2)$ while avoiding arithmetic in $GF(p^6)$.

2.2.4 Elliptic Curve Cryptosystem

In 1985 Neal Koblitz [45, 46] and Victor Miller [65] independently proposed ECC using the group of points on an elliptic curve defined over a finite field in discrete logarithm cryptographic systems. The primary advantage that elliptic curve systems over multiplicative group of finite field is the absence of a sub-exponential time algorithm that could find discrete logarithms in these groups. Consequently, one can use an elliptic curve group that is smaller in size while maintaining the same level of security. The necessary condition for the security of all elliptic curve cryptographic schemes is that the ECDLP is intractable. Similarly like DLP, no proof that the ECDLP is indeed a hard problem. evidence for its hardness has been gathered over the years. First, the problem has been extensively studied by researchers for the last 16 years and no general-purpose sub-exponential time algorithm has been discovered. An elliptic curve $E$ over the field $F$ is a smooth curve in the so called "long Weierstrass form"

$$Y^2 = +a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6$$

We let $E(F)$ denote the set of points $(x, y) \in F^2$ that satisfy this equation along with a point at infinity denoted as $O$.

Two finite fields are of particular interest. The finite field $F_p$ and the finite field
Elliptic curve is a new emerging class of public key cryptosystems that may successfully compete in the future with the current monopoly of the RSA cryptosystem. If the elliptic curve is chosen correctly, the best known algorithm for finding the discrete logarithm is of exponential difficulty. The cryptographic keys may be significantly shorter for elliptic curve cryptosystems compared to RSA, which results in faster and more compact implementations. The other advantage of using elliptic curves for constructing cryptosystems is that each user may (but does not need to) choose his/her own curve. If this is the case, reconstructing the private key of one user, does not provide any information that could be used to reconstruct the private key of another user. According to the current knowledge, choosing an appropriate elliptic curve is equivalent to calculating the number of points on the elliptic curve, and checking whether this number has a large prime divisor. Several methods have been devised for choosing an appropriate elliptic curve. The anex to the IEEE 1363 standard lists three of them: counting number of points on a randomly chosen curve using Schoof’s algorithm, constructing the curve using the Weil theorem, and constructing the curve using the method of complex multiplication.

Selection of parameters

The system parameters, such as the elliptic curve group of order $n$ with $p$ as the largest prime order factor, the fixed point $P$ of order $p$ to be chosen as follows:-

- Two constants $a$ and $b$ are selected randomly, such that $4a^3 + 27b^2 \neq 0$

- The order of the elliptic curve ($n$) to be computed by using the Schoof’s algorithm or Weil’s theorem based on the type of the curve

- The fixed point $P$ of order $p$ is chosen
**ECC-Diffie-Hellman**

The Elliptic curve cryptosystem is implemented by using the Diffie-Hellman protocol for exchanging the session key. Later, the symmetric key cryptosystem such as DES, IDEA to name a few is used for encryption by using the session key. The procedure is discussed below.

1. Alice selects a random integer $m$ and computes $Q = mP$ and sends this $Q$ to Bob.

2. Bob selects a random integer $n$ and computes $R = nP$ and sends this $R$ to Alice.

3. Alice computes $X = mR = mnP$, thus the session key is $X$.

4. Bob computes $X = nQ = nmP$, thus the session key is $X$.

The session key $X$ is used as a secrete key for encryption using one of the symmetric key cryptosystems. Another option for encryption is by using the El-Gamal encryption scheme.

### 2.3 Attacks on DLP based public key cryptography

From the above discussion, it is observed that the DLP is the basis for many popular cryptosystems. Thus the present study deals with the cryptanalysis of the DLP. In this section the methods to improve the computation of DLP are discussed.

#### 2.3.1 Improving the computation of DLP using the number theoretic approach

As discussed in the introduction chapter the computation of DLP can be improved through three approaches. The following section briefs the attacks developed to solve the DLP by using the first approach such as through the number theoretic properties.
Generic attacks on any group structure

The following attacks can work on any group structure. These types of attacks are popularly known as square root attacks, as they need $O(\sqrt{n})$ group operations, where $n$ is the order of the generator.

**Shanks Baby Step and Giant Step Algorithm** The algorithm is originally developed by D.Shanks on 1971 [85, 44, 88, 90, 91]. This algorithm requires the construction of two arrays of group elements as follows:-

**Giant step** is defined by

$$S = \{(i, g^i)^{\lceil \sqrt{n} \rceil} | i = 0 \ldots \lceil \sqrt{n} \rceil \}$$

**Baby step** is defined by

$$T = \{(j, y \times g^j) | j = 0 \ldots \lceil \sqrt{n} \rceil \}$$

To compute the discrete logarithm, find group elements that appear in both the list. Thus the logarithm is

$$\log_g y \equiv i\lceil \sqrt{n} \rceil - j \mod n$$

**Pollard-Rho Method** This is the popular algorithm proposed by J.Pollard on 1978. The Pollard-Rho works by first defining a pseudo-random sequence of elements from a group and then looking for a cycle to appear in the sequence [73].

The sequence can be defined by

$$\begin{cases} 
  y \times Y_i & : Y_i \in S_1 \\
  Y_i^2 & : Y_i \in S_2 \\
  g \times Y_i & : Y_i \in S_3 
\end{cases}$$

Where $S_1, S_2, S_3$ are an arbitrary partition of the group into roughly equal sized sets.

The procedure is as follows.
1. Find $a_i, b_i$ at random and compute $Y = g^{a_i} y^{b_i}$
2. Find the next sequence by using the equation given above
3. At one point $Z_{i-1} = Z_i$
4. Find $x$ such that $x = (a_i - k)(b_i - l)^{-1} \mod n$ (i.e.) $g^{a_i} y^{b_i} = g^k y^l \mod n$.

**Sub-exponential time algorithms**

The Index Calculus Methods are the most prominent collection of algorithms that have successfully used additional knowledge of the underlying groups to provide sub-exponential algorithms. The basic idea, which goes back to Kraitchik [60] is that if

$$
\prod_{i=1}^{m} x_i = \prod_{j=1}^{n} y_i \quad (2.1)
$$

for some elements of $GF(q)^*$, then

$$
\sum_{i=1}^{m} \log_g x_i \equiv \sum_{j=1}^{n} \log_g y_j \pmod{q-1} \quad (2.2)
$$

If we obtain many equations of the above form, and they do not involve too many $x_i$ and $y_i$, then the system can be solved. This is similar to the situation in integer factorization, discussed greater detail in [51], in which one needs to find a linear dependency among a system of linear equation modulo 2. For more details on index calculus methods for discrete logarithms refer [78]. Progress in index calculus algorithms has come from better ways of producing relations that lead to equations such as 2.2. However even with the primitive method known as Random Method one can obtain running time bounds of the form

$$
\exp\left( (c + O(1))(\log p)^{\frac{2}{3}}(\log \log p)^{\frac{1}{3}} \right) \quad (2.3)
$$

for some constant $c$.

The algorithm has two steps

- A pre-computation step where the logarithms of $\log_g b$ of all members of the factor base is obtained.

- A computation step, which tries enough $g^a y$ until the result factors over the factor base, thus providing the requested logarithm $\log_g y$[78, 89].
The pre-computation step is computationally the more expensive step, and it has two phases

- First phase is to find the linear relations relating the logarithms of the primes in the factor base.
- Second phase is to solve this linear system using techniques from linear algebra.

The general algorithm is the primitive method of ICM and it is described below[18].

**General algorithm**

INPUT a generator $g$ of a cyclic group $G$ of order $n$ and an element $y$

OUTPUT $\log_g y$.

- Select a factor base $FB=\{p_1, p_2, ..., p_t\}$, which belongs to $G$ such that a significant portion of elements of $G$ can be efficiently expressed as a products of elements from $FB$.

- Find a linear system using the procedure as given below
  * Select a random integer $k$, such that $0 \leq k \leq n - 1$ and compute $g^k$
  * Try to write $g^k$ as a product of elements in $FB$ as
    \[ g^k = \prod_{i=1}^{t} p_i^{c_i}, \ c_i > 0, \quad (2.4) \]
    
    for any $k$. Then, $k \equiv \sum_{i=1}^{t} c_i \log_g p_i$
  * Repeat the above steps to get the value of $t+c$ equations.

- Solve this linear system to obtain $\log_g p_i$

- Compute $\log_g y$
  * Select a random integer, $k$, $(0 \leq k \leq n - 1)$ and compute $yg^k$
  * Try to write $yg^k$ as a product of elements in $FB$
\begin{equation}
yg^k = \prod_{i=1}^{t} p_i^{d_i},
\end{equation}

for any \( k \). Then, \( \log_g y = (\sum_{i=1}^{t} d_i \log_g p_i - k) \mod n \)

The very first analysis of the asymptotic running time of Index calculus algorithms appeared in the 1970s and were of the form 2.3. All the progress in the 1970s and 1980s was in obtaining better values of \( c \), and for a long time \( c = 1 \) was the record, both for discrete logs modulo primes and for integer factorization. Coppersmith and Odlyzko [25] presented three versions of index calculus method in 1986. Later LaMacchia and Odlyzko [49] reported the implementation of two of these three versions namely, linear sieve and Gaussian integer methods in 1991. An implementation of cubic sieve method is reported by Abhijit Das and Veni Madhavan in 2005 [2]. For fields \( GF(q) \) with \( q = p^n \) for small \( p \), Coppersmith’s algorithm [22] offered running time of the form

\begin{equation}
\exp((C - o(1))) (\log q)^{\frac{1}{3}} (\log \log q)^{\frac{2}{3}}
\end{equation}

for a positive constant \( C \) [68]. For some fields \( GF(q) \) with \( q = p^n \) in which both \( p \) and \( n \) grow even bounds of the first form were not available. This lack of progress led to fairly wide speculation that running times for Integer factorization and for discrete logs in prime fields could not be improved beyond 2.3 with \( c = 1 \).

However in 1988 Pollard found a new approach for factoring integers. This method was developed into the Special Number Field Sieve (SNFS) by Hendrik Lenstra, and later into the General Number Field Sieve (GNFS) through a collaboration of several researchers. Initially there was wide skepticism as to whether this method would be practical, but those doubts have been dispelled. The first version of the GNFS for discrete logs was developed by [34]. Gordon’s algorithm was improved by [76, 77], later Weber efficiently implemented the algorithm [99, 100]. Adleman [3] has invented the function field sieve, which can be regarded as a generalization and often an improvement of the Coppersmith algorithm for fields of small characteristic. As a result, we now posses a variety of discrete log algorithms with running time of the form 2.6. For fields \( GF(q) \) with \( q = p^n \), where \( n \) is large, the running time bound of 2.6 holds with \( C = 1.5262 \ldots \). For \( n \) small,
in general we know only that 2.6 holds with $C = 1.9229\ldots$ for special primes $p$, which initially were just the primes of the Cunningham form with $p = r^n + a$, where $r$ and $a$ are small runs in times of the form 2.6 with $C = 1.5262$ or even less using NFS [82, 83, 84].

Sub-exponential index calculus algorithms have been developed for a variety of discrete log problems. All algorithms for discrete logs that are claimed to run in time of the form 2.6 for some constants $C$ are heuristic, in that there is no proof they will run that fast. If one willing to settle for running times of the form 2.3, then it is possible to obtain rigorous probabilistic algorithms.

**Smoothness** The main concept associated with ICM is smoothness property of integers. The smoothness property of integers is the base for sub-exponential time algorithms to solve the DLP. An element is called as $y$ smooth, if it has no prime divisor larger than some bound $y$. The distribution of smooth integers is studied extensively [40, 39, 41]. Berstein presented a tight bounds on the distribution of smooth integers [8], a linear time algorithm to list $y$-smooth integer up to $x$ and smooth part of integers [9, 10] and several algorithms for number of integers free of large prime factors [10]. The results on smoothness of integers are available [40], smoothness on algebraic integers in [17] and smoothness of polynomials are in [70]. Smoothness estimates of this type are also crucial for the few rigorous proofs of running times of probabilistic algorithms.

**Linear systems over finite fields** Index calculus algorithms require solutions of large sets of linear equations over finite fields. For a long time in the 1970s and early 1980s this step was regarded as a major bottleneck, affecting the asymptotic running time estimates of algorithms such as the continued fraction method and the quadratic sieve. Fortunately the linear systems of equations produced by all index calculus algorithms are sparse. This makes possible development of algorithms that take advantage of this sparsity and operate faster than general ones. The introduction of Structured Gaussian elimination [48, 67] and of the finite field versions of the Lanczos and conjugate gradient algorithms [25, 67, 48], and the subsequent discovery of the Wiedemann [102] algorithm led to a reduction in the
estimate of the difficulty of the equation solving phase. However, practice lagged behind the theory for a long time. The main advances in linear algebra for index calculus algorithms in the 1990s came from the parallelization of the Lanczos and Wiedemann algorithms by Coppersmith [23, 24]. Currently the most widely used parallel method is Montgomery’s [66] version of the Lanczos algorithm, where it is used after structured Gaussian elimination reduces the matrix to manageable size. These parallelization methods essentially speed up the basic algorithms over the field of two elements by factors of 32 or 64 and are very effective. There are concerns that linear equations might be done on a network of distributed machines, each with modest memory requirements and minor communication needs, linear equation solutions require a closely coupled system of processors with a large memory. Still, those requirements are not too onerous [69].

In discrete logarithm, the linear algebra is a more serious problem, since solutions have to be carried out not modulo 2, but modulo large primes. Hence the parallelizations of Coppersmith and Montgomery do not provide any relief, and the original structured Gaussian elimination, Lanczos, and conjugate gradient methods as implemented in [48] are still close to best possible. Also, Lambert reported some careful analysis and improvements [50]. The table 2.1 presents the problems solved in ICM using various methods. The algorithms reported in the table used

<table>
<thead>
<tr>
<th>Method</th>
<th>First phase</th>
<th>Second phase</th>
<th>Problem size</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear sieve</td>
<td>√</td>
<td>√</td>
<td>192bits</td>
<td>1992</td>
</tr>
<tr>
<td>Gaussian integer</td>
<td>√</td>
<td>√</td>
<td>200bits</td>
<td>2000</td>
</tr>
<tr>
<td>Cubic sieve</td>
<td>√</td>
<td>—</td>
<td>150bits</td>
<td>2005</td>
</tr>
<tr>
<td>Number Field Sieve (special primes)</td>
<td>√</td>
<td>√</td>
<td>129digits</td>
<td></td>
</tr>
<tr>
<td>Number Field Sieve</td>
<td>√</td>
<td>√</td>
<td>120 digits</td>
<td>2002</td>
</tr>
<tr>
<td>Number Field Sieve on $GF(2^n)$</td>
<td>√</td>
<td>√</td>
<td>$2^{607}$</td>
<td>2002</td>
</tr>
</tbody>
</table>
one of the methods such as linear sieve, Gaussian integer, cubic sieve and Number Field Sieve in the first phase and one of the linear algebra techniques such as Lanczos, conjugate gradient and Wiedemann methods in the second phase.

From the above discussions, it is observed that the popular methods for generating linear relations are linear sieve, cubic sieve, Gaussian integer and Number Field Sieve. The linear sieve method is considered in the present study. Generally, the generation of relation is highly dependent on the smoothness of elements in the group. The smoothness in turn depends on the size of the factor base. Consequently the number of relations to be generated depends on the size of factor base and the sieve length i.e., the range of numbers to be checked for smoothness. These two characteristics, namely size of factor base and the sieve length are studied and reported in the literature. Next, reducing the number of relations also studied by LaMacchia and Odlyzko through the experimental work on structured Gaussian elimination for a problem of size around 192 bits, in which it is mentioned that the linear system ought to be sparse and there should be considerable number of relations than unknowns to get better reduction [48]. Recently Roberto Avanzi proposed a new filtering technique called as harvesting method to reduce the size of the linear system, which removes duplicate equations and singletons from the system. He also claimed that it can improve the performance of ICM by more than 30% [5]. Finally the popular linear algebra techniques for solving the reduced linear systems are Lanczos, conjugate gradient and Wiedemann.

**Some special algorithms**

In this section we discuss briefly some algorithms that apparently do not work very well. In a field $GF(p)$, Well’s [101] has shown that for any $u$, $1 \leq u \leq p - 1$, if $g$ is a primitive root modulo $p$, then one can write

$$\log_g u \equiv \sum_{f=1}^{p-2} (1 - g^f)^{-1} u^f \pmod{p}$$

This was not designed as an algorithm at all. The Herlestad-Johannesson [38] method was designed to work over the field $GF(2^n)$, and was reported by those authors to work efficiently for fields as large as $GF(2^{31})$. However, the heuristics
used by those authors in arguing that the method ought to work efficiently in larger fields as well as to be very questionable. As usual, \( GF(2^n) \) is represented as polynomials over \( GF(2) \) modulo some fixed irreducible polynomial \( f(x) \) of degree \( n \) over \( GF(2) \). In order to compute the logarithm of \( h(x) \) to base \( x \). Herlestam and Johannesson proposed to apply a combination of the transformations.

\[
h(x) \leftarrow h(x)^{2^r}, h(x) \leftarrow x^{-2^s}h(x)\]

so as to minimize the Hamming weight of the resulting polynomial, and apply this procedure iteratively until an element of low weight, for which the logarithm was known, was reached. There is no reason to expect such a strategy to work, and considerable numerical evidence has been collected which shows that this method is not efficient [14], and is not much better than a Pollard-random walk through the field. However some unusual phenomena related to the algorithm have been found whose significance is not yet understood. Another approach to computing discrete logarithms in fields \( GF(2^n) \) was taken by Arazi [68]. He noted that if one can determine the parity of the discrete logarithm of \( u \), then one can quickly determine the discrete logarithm itself. Arazi showed that one can determine the parity of discrete logarithms to base \( g \) fast if \( g \) satisfies some rather complicated conditions. Since being able to compute discrete logarithms to one base enables one to compute them to any other base about equally fast. However, so far no algorithm has been found for finding such primitive elements \( g \) in large fields \( GF(2^n) \), nor even a proof that any such elements exists [68].

**Attacks on ECC**

**Generic attacks** All generic attacks such as Shanks baby step and giant step, Pollard-Rho method and Pohlig-Hellman methods are applicable to solve ECDLP.

**Attacks using group structure of elliptic curve** This section presents the attacks developed based on the structure of the elliptic curve group. The popular attacks of this kind are MOV, Weil-Descent and anomalous attack to name a few. The MOV attack depends on Weil-pairing or Tate-pairing.
Weil-Pairing

Weil-pairing is defined as follows
e_n : E[n] \times E[n] \to \mu_n \text{ where } E[n] \text{ is set of } n\text{-torsion points such that } \{ P \in E(k)[n] | P = O \} E[n] \cong \mathbb{Z}/n\mathbb{Z} \oplus \mathbb{Z}/n\mathbb{Z}, \mu_n = \{ \alpha | \alpha^n = 1, \alpha \in \overline{k} \} \text{ If } P \text{ and } Q \text{ are } n\text{-torsion points then } e_n(P, Q) \text{ is an } n^{th} \text{ root of unity in } \overline{k} \nongrav bronk

e_n(P, Q) = \begin{cases} 1 & : \ P = Q \\ \frac{f_p(Q)}{f_Q(P)} & : \text{ otherwise} \end{cases}

The properties of Weil pairing are interesting

(1) Bilinear: For \( P_1, P_2, Q_1, Q_2 \in E[n] \)

\[ e_n(P_1 \oplus P_2, Q_1) = e_n(P_1, Q_1) \cdot e_n(P_2, Q_2) \]

\[ e_n(P_1, Q_1 \oplus Q_2) = e_n(P_1, Q_1) \cdot e_n(P_2, Q_2) \]

(2) Non degenerate: If \( e_n(P, Q) = 1 \ \forall Q \text{ then } P = O. \)

Similarly, the Tate-pairing is a map of \( e_n : E(F_q)[n] \times E(F_{q^k})[n] \to F_{q^k}^* \)

MOV Attack

Consider \( E/F_p \) and \( |E(F_p)| = l \text{ prime and therefore } E(F_p) \leq E[l] \) as subgroup.
\( E[l] = \mathbb{Z}/l\mathbb{Z} \oplus \mathbb{Z}/l\mathbb{Z} \) and \( |E[l]| = l^2; E[l] = E(F_p) \oplus \mathbb{Z}/l\mathbb{Z}.Q \) for some \( Q \in E[l] \)

The Elliptic Curve Discrete Logarithm Problem is, given \( T \in E(F_p) \) to compute \( m \) such that \( T = mS \). The attack is

1. Compute \( e_l(S, Q) \)
2. Compute \( e_l(T, Q) = e_l(mS, Q) = e_l(S, Q)^m \)

by bilinearity property this become
\( a = e_l(T, Q) = e_l(mS, Q) = e_l(S, Q)^m \)

Suppose \( e_l(S, Q) = b \) then \( a = b^m; a = \mu_l^m \)

\( m \) can be calculated if discrete logarithm can be calculated in \( \mu_l \leq F_p \). Actually an extension of \( F_p \) that contain \( \mu_l \) will do. So we find \( k \) such that \( F_{p^k} \) contain \( \mu_l \)
i.e we need \( l \) to divide \( |F_{p^k}^*| = p^k - 1 \) \[62\].

Anomalous Attack

This attack is possible on the anomalous curve (i.e) \(|#E(F_p)| = p. \) Here the attack on super anomalous curve is presented. Super anomalous curve is an
extension of anomalous curve. Satoh-Araki-Smart algorithm used to solve DLP over super anomalous elliptic curve is explained below [75].

Super anomalous elliptic curve over a ring $\mathbb{Z}/n\mathbb{Z}(n = \prod_{i=1}^{k} p_i^{e_i})$ is defined by extending anomalous elliptic curve over a prime field $\mathbb{F}_p$. They have $n$ points over a ring $\mathbb{Z}/n\mathbb{Z}$ and $p_i$ points over $\mathbb{F}_p$, $\forall p_i$

The Elliptic curve $\tilde{E}(F_p)$ is the set $(x, y)$ on the curve $\tilde{E} = y^2 = x^3 + a_p x + b_p (\mod p)$ including a point at infinity $Q_p$.

If $\tilde{E}(F_p) = p$ (anomalous) then $\tilde{E}^*(F_p) \in \tilde{E}(F_p) - \{Q_p\}$

The lifting curve $E = y^2 \equiv x^3 + ax + b$ is an elliptic curve satisfying $a \equiv a_p (\mod p)$ and $b \equiv b_p (\mod p)$ and $a_p, b_p \in \mathbb{F}_p$ and $a_p, b_p \in \mathbb{Z}$ or $\mathbb{Z}/p^2\mathbb{Z}$.

A point $(x, y) \in E(F_p)$ or $E(\mathbb{Z}/p^2\mathbb{Z})$ satisfying $x \equiv x_p (\mod p)$ and $y \equiv y_p (\mod p)$

Elliptic curve over $\mathbb{Z}/n\mathbb{Z}$

Let the composite $n = \prod_{i=1}^{k} p_i^{e_i}$, $k$ is the number of distinct prime factors of $n$ and $E$ be an elliptic curve $E = y^2 \equiv x^3 + ax + b$. A group $E(\mathbb{Z}/n\mathbb{Z})$ is defined as the direct sum of $k$ groups (i.e) $E(\mathbb{Z}/n\mathbb{Z}) = \bigoplus_{i=1}^{k} E(\mathbb{Z}/p_i^{e_i}\mathbb{Z})$

The other popular attacks are Weil-descent attack [42] and faster attack on ECC using Pollard-Rho [92, 93, 94].

**Experimental results on ECC attacks**

Certicom [19] initiated an ECDLP challenge in November 1997 in order to encourage and simulate research on ECDLP. Their challenges consist of instances of ECDLP on a selection of elliptic curves. The challenge curves are divided into three categories listed below. In the following, $ECC_p - k$ denotes a randomly selected elliptic curve over a field $F_p$. $ECC2 - k$ denotes a randomly selected elliptic curve over a field $F_{2^m}$, and $ECC2K - k$ denotes a Koblitz curve over $F_{2^m}$. In all cases, the bit size of the order of the underlying finite field is equal or slightly greater than $k$.

- Randomly generated curves over $F_p$, where $p$ is prime: $ECC_p - 79, ECC_p - 89, ECC_p - 97, ECC_p - 109, ECC_p - 131, ECC_p - 163, ECC_p - 191, ECC_p - 239, ECC_p - 359$
Randomly generated curves over $F_{2^m}$, where $m$ is prime: $ECC2-79$, $ECC2-89$, $ECC2-97$, $ECC2-109$, $ECC2-131$, $ECC2-163$, $ECC2-191$, $ECC2-238$ and $ECC2-353$.

Koblitz curves over $F_{2^m}$, where $m$ is prime: $ECC2K-95$, $ECC2K-108$, $ECC2K-130$, $ECC2K-163$, $ECC2K-238$ and $ECC2K-358$.

Escott et al. [31] reported on their 1998 implementation of parallelized Pollard’s rho algorithm which incorporates some improvements of Teske [92]. The hardest instance of the ECDLP they solved was the Certicom $ECC_p-97$ challenge. For this task they utilized over 1200 machines from at least 16 countries, and found the answer in 53 days. The following challenges are solved

- $ECC_p-79$, $ECC_p-89$, $ECC_p-97$
- $ECC2-79$, $ECC2-89$, $ECC2-97$
- $ECC2K-95$, $ECC2K-108$

Recently Menezes et al. [64] solved for any instance of the ECDLP over any elliptic curve on the week fields. They showed that the ECDLP can be solved significantly in less time using their method in comparison with the Pollard-Rho method to solve the hardest instances. Solving the ECDLP using the ICM is an open problem, but recently the DLP in hyper elliptic curves are solved using this method [5].

### 2.3.2 Improving the DLP using the computational approach

The computation of DLP can be improved by using efficient implementation of the traditional methods. The above algorithms solves the DLP in exponential and sub-exponential time. The running time of the Shanks and Pollard algorithms have not been improved to any substantial extent. Only improvements by constant factors have been obtained [72, 92, 98]. There has been progress, on the other hand, in obtaining fast parallel versions in which the elapsed time for the computation shrinks by a factor that is linear in the number of processors used [35, 72, 98].
However, the basic processing for any of these algorithms still requires a total of about $p^{1/2}$ steps, where $p$ is the largest prime dividing the order of $g$. The lack of progress in several decades is very important. Many modern public key cryptosystems based on discrete logarithms, such as the U.S Digital Signature Algorithm (DSA) [63, 79], rely on the Schnorr method [80], which reduces the computational burden normally imposed by having to work in a large finite field by working within a large multiplicative subgroup $Q$ of prime order $q$. The assumption is that the discrete log problem in $Q$ cannot be solved much faster than $q^{1/2}$ steps. For $q$ of order $2^{160}$, as in DSA, this is about $10^{24}$ group operations. A mips-year is equivalent to about $3.10^{13}$ instructions, so breaking DSA, say, with the Pollard or Shanks algorithms would require over $10^{12}$ mips-year, which appears to be adequate for a while at least.

From the ICM point of view, the implementation of linear sieve and Gaussian integer method proposed by LaMacchia and Odlyzko is an example of improving the DLP though the computational approach. In the same line the cubic sieve method is implemented by Abhijit Das and Veni Madhavan. Further the Number Field Sieve method is implemented and improved though computationally [76, 77]. Through, a new filtering technique the ICM is improved by 30% [5]. The technique is used to reduce the large linear system generated in the first step of ICM to smaller system for the solving step.

From ECDLP perspective, the parallel collision search on binary anomalous curves is one way of improving the computation of DLP. The equivalence classes derived from the anomalous curve aid in improving the Pollard-Lambda method [33]. This allows to perform the parallel collision search to solve the ECDLP.

### 2.3.3 Improving the computation of DLP using the structure of group, exponents and order of other elements related to DLP

The algorithms developed to solve the DLP with additional information other than $g$ and $y$ such as the order of the group, the exponents and other details regarding the DLP are reported. The efficient algorithm to solve the DLP, when the factors
of $p - 1$ are known and small is Pohlig-Hellman method.

**Pohlig-Hellman Method**

This is a popular algorithm introduced by Pohlig-Hellman on 1978. If the order of the group is known along with the complete factorization and the factors are relatively small then this attack is possible [71].

Let $p - 1 = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$ and $g$ be the generator of order $p - 1$. Then

$$g^x = y \mod p \Rightarrow$$

$$\alpha^x = \beta \mod p \text{ where } \alpha = g^{(p_1^{-1} \mod p_{1}^{e_1})} g^{(p_2^{-1} \mod p_{2}^{e_2})} \cdots g^{(p_k^{-1} \mod p_{k}^{e_k})}$$

and

$$\beta = y^{(p_1^{-1} \mod p_{1}^{e_1})} y^{(p_2^{-1} \mod p_{2}^{e_2})} \cdots y^{(p_k^{-1} \mod p_{k}^{e_k})}$$

The logarithm in the small subgroups are solved by using one of the popular square root algorithms. Later the Chinese remainder method is used to combine the results $x_i \mod p_i^{e_i}$ to retrieve $x \mod p - 1$.

**Other attacks**

Apart from the above algorithms to solve the DLP, it is worth noticing the other popular attacks on this hard problem such as timing attacks, attacks on short exponents, attacks on prime order subgroup, malleability attack, man in the middle attack, man in the meet attack, insider’s attacks, outsider’s attack to name a few. Some of the above attacks are designed on the vulnerabilities of the selection of parameters for the protocols based on this hard problem, such as the generation of $p$, $q$ and $g$. Minding $p$’s and $q$’s by R. Anderson and S.Vaudenay addresses the selection of these parameters and the vulnerabilities [4, 96].

The exponent selection is also a crucial point in the implementation of protocols. This may leads to give a way to powerful attacks on the systems. One of the attacks of this kind is the attack on short exponents by Van Oorschot’s and Wiener [97]. In the current technology, it is considered as infeasible to compute
the DLP in a group of order $\approx 1024$ bits. Van Oorschot and Wiener examined the difficulty of computing the DLP for an exponent of size $\approx 160$ combined with the random prime $p$ of size $\approx 1024$ bits. They suggested the use of prime order subgroup along with the short exponents or the use of safe primes. Lim and Lee [56] have shown a specific interest in investigating the computation of the DLP in a prime order subgroup of order $\approx 160$ bits of $p$ of size $\approx 1024$ bits by extracting $x \bmod \Omega(\beta)$, where $\beta$ is the product of elements of smooth order and the prime $p$ is assumed to be of random and the $p - 1$ has many small factors apart from the large one with 160 bits. In both the cases random primes are used for the computations and assumed to have many small factors apart from the large one for $p - 1$. The difference with respect to the structure of $p - 1$ is that, the former solved the problems of generators of order $|p - 1|$ and the later solved the problems of generators of order $|q|$ and computations are restricted to prime order subgroup $q$.

Further Dan Boneh, Antoine Joux, Q. Phong and Nguyen reported an attack on Textbook El-Gamal and RSA encryption on the encryption of messages of smaller size [16]. Since the encryption procedure followed in the above two cryptosystems are simple operations such as exponentiation in RSA and multiplications or XOR in El-Gamal, the above attack breaks these systems, when the smaller size messages are used in the encryption, for example a session key to be encrypted using the above algorithms.

The chosen-ciphertext attack is a popular attack on DLP based schemes to recover the plaintext. The additional information used in this type of attack is the chosen ciphertext.

**Chosen-ciphertext attack** A chosen-ciphertext attack (CCA) is an attack model for cryptanalysis in which the cryptanalyst gathers information, at least in part, by choosing a ciphertext and obtaining its decryption under an unknown key. A number of otherwise secure schemes can be defeated under chosen-ciphertext attack. For example, the El-Gamal cryptosystem is semantically secure under chosen-plaintext attack, but this semantic security can be trivially defeated under a chosen-ciphertext attack. When a cryptosystem is vulnerable to chosen-ciphertext attack, implementers must be careful to avoid situations in which an
adversary might be able to decrypt chosen-ciphertexts (i.e., avoid providing a decryption oracle). For example, given an encryption \((c_1, c_2)\) of some (possibly unknown) message \(m\), one can easily construct a valid encryption \((c_1, 2c_2)\) of the message \(2m\). To achieve chosen-ciphertext security, the scheme must be further modified, or an appropriate padding scheme must be used. The other scheme related to El-Gamal i.e., Cramer Shoup resists the chosen cipher text attacks \([27, 6, 7]\).

Timing attacks are another class of attacks, based on the computational time required by the prominent operations of the DLP based schemes.

**Timing attacks**

Timing attacks attempt to exploit the variations in computational time for private key operations to guess the private key. This type of attack is primitive in the sense that no specialized equipment is needed. An attacker can break a smart card key by simply measuring the computational time required by the card to respond to user inputs and recording those user inputs. The viability of this attack is important to any smart card implementation using vulnerable cryptosystems. An attacker with prolonged passive eavesdropping ability may be able to break the private key and gain access to the information stored on the card. This will give the attacker access to sensitive information.

**Timing Attack on ECC**

The timing attack requires to find a partition of an input message set into groups that require different amounts of computational time depending on the key bits. For a timing attack to work, there must be some predictable variation in computational time dependent upon the input messages. The variations in computing times for multiplications and inverses allows to predict the key bits in an implementation of El-Gamal cipher in ECC. These two steps are the most time consuming steps in the process of adding two elliptic curve points or doubling an elliptic curve point. The implementation of these operations in ECC may leads a way to timing attacks.
2.4 Summary

In this chapter, we reviewed the contributions of researchers, specifically in terms of solving the DLP. This helps in identifying the importance of cryptanalysis on the DLP. Solving DLP is viewed by the researchers from various perspectives. The present study improves the computation of DLP using the third approach as discussed earlier. The approach exploits the additional information regarding the DLP other than $g$ and $y$. In general, the methods developed by using the above approach help the cryptanalyst to create a trap door to break the system and for the cryptographer to built a secure system from the vulnerabilities caused by the trap doors. In particular, the computation of DLP is improved through the traditional ICM, specialized ICM and new methods. The ICM is the most effective attack on solving the DLP. The key parameters of ICM such as the size of the factor base and sieve length are studied from generation of relations perspective. The reduction in the linear system from larger to smaller system, which makes the second phase efficient is also a key factor for the overall performance of algorithm. Thus in the present study, the reduction, size of the factor base and sieve length are studied all together as a group from the perspective of both the generation and solving linear systems phases of ICM.

The ICM is studied so far as a general method for prime fields and for special type of primes using number field sieve. In the present study, the basic ICM is modified using the combination of Pohlig-Hellman and traditional ICM for some instances of primes to recover the ephemeral keys. Ephemeral keys are used in each session between the communicators in the popular DLP based cryptosystems. A new smoothness concept named as smooth numbers over $Z_p^*$ is defined in the present study and the DLP is solved efficiently in reduced time using various techniques based on the above new smooth concept. Furthermore, the Random method of ICM and the third step (individual logarithm step) of ICM are improved through the characteristics of smooth numbers over $Z_p^*$. From the ECDLP point of view, Pollard family of algorithms are the most effective algorithms. In the present work the experiments are conducted on the finite field of $E_{a,b}(F_{2^m})$. 

40