Journal of Information Processing Systems, Vol.6, No.3, September 2010 DOI : 10.3745/JIPS.2010.6.3.335

Solving the Discrete Logarithm Problem for Ephemeral Keys in Chang and Chang Password Key Exchange Protocol

R. Padmavathy* and Chakravarthy Bhagvati**

Abstract—The present study investigates the difficulty of solving the mathematical problem, namely the DLP (Discrete Logarithm Problem) for ephemeral keys. The DLP is the basis for many public key cryptosystems. The ephemeral keys are used in such systems to ensure security. The DLP defined on a prime field Z^*_p of random prime is considered in the present study. The most effective method to solve the DLP is the ICM (Index Calculus Method). In the present study, an efficient way of computing the DLP for ephemeral keys by using a new variant of the ICM when the factors of $p-1$ are known and small is proposed. The ICM has two steps, a pre-computation and an individual logarithm computation. The pre-computation step is to compute the logarithms of a subset of a group and the individual logarithm step is to find the DLP using the pre-computed logarithms. Since the ephemeral keys are dynamic and change for every session, once the logarithms of a subset of a group are known, the DLP for the ephemeral key can be obtained using the individual logarithm step. Therefore, an efficient way of solving the individual logarithm step based on the newly proposed pre-computation method is presented and the performance is analyzed using a comprehensive set of experiments. The ephemeral keys are also solved by using other methods, which are efficient on random primes, such as the Pohlig-Hellman method, the Van Oorschot method and the traditional individual logarithm step. The results are compared with the newly proposed individual logarithm step of the ICM. Also, the DLP of ephemeral keys used in a popular password key exchange protocol known as Chang and Chang are computed and reported to launch key recovery attack.

Keywords—Ephemeral Key, Pohlig-Hellman Method, Van-Oorschot Method, Index Calculus Method, Chang-Chang Password Key Exchange Protocol

1. INTRODUCTION

This paper investigates the methods of solving the Discrete Logarithm Problem (DLP) for ephemeral keys. The ephemeral keys may be unique for each session or they may be reused for different sessions of the same party. For example, the ANSI X9.42 standard, which specifies several Diffie-Hellman protocols states that an ephemeral key is a private or public key that is unique for each execution of cryptographic schemes”. Other protocols do not place any restrictions on the reuse of ephemeral keys [9]. The ephemeral keys, which are unique for each session,

Manuscript received March 8, 2010; accepted August 7, 2010.

Corresponding Author: R.Padmavathy
* Dept. of Computer Science and Engineering, National Institute of Technology, Warangal, INDIA
** Dept. of Computer and Information Sciences, University of Hyderabad, Hyderabad, INDIA

are considered in the present study.

The ephemeral keys, which are used in the Discrete Logarithm based public key cryptosystems, are to ensure the security of the systems. One way of retrieving this key is to solve the mathematically hard problems such as the Discrete Logarithm Problem (DLP). The ephemeral keys are dynamic and change for every session between Alice and Bob while the static keys remain the same and live longer. Since the life time of ephemeral keys is short, it is hard to recover these keys within the short span of time by using the attacks with the target of solving the DLP. Thus the problem of solving the DLP for retrieving the ephemeral keys is formulated.

Let a group *(G, *)* consists of a set *G* and a binary operation *. The order of an element, say a, of a finite group *G* is defined to be with the smallest value *t* such that $a^t = 1$. Some of the well known groups used in the cryptography are the set $Z^*{}_p$ with multiplication *mod p* (*p* is a prime), the multiplication group of the field F_{2^m} and the addition group formed by the collection of points defined by an elliptic curve over the finite field. For a given prime p, a generator $g \in Z_p^*$ and an element $y \in Z_p^*$, the problem of finding x, in the range of $0 \le x \le p-2$, such that $g^x = y (\bmod p)$, is known as the DLP. Some of the attacks on the DLP are discussed below.

The methods to solve the DLP on $Z^*{}_P$ can be divided into two types-- the general-purpose and special-purpose methods. The general purpose algorithms include, Shanks baby step - giant step, the Pollard family of algorithms and the Index Calculus Methods. The following paragraphs describe the general-purpose methods. Apart from the exhaustive search to solve the DLP a well known deterministic algorithm is Shank's baby step- giant step algorithm. It requires $o(\sqrt{n})$ group operations and space [7]. The Pollard Rho method, which is a probabilistic one, has similar square root running time but avoids large space requirements [11].

The DLP can be computed in sub exponential time using the ICM, if there is more structure to the group beyond the set of elements and the group operation. Specifically, certain group elements can be labeled as smooth, when it can be factored into a product of group elements from some relatively small factor base. The ICM uses a fixed small set called the factor base B and tries to write elements as a product of members of the factor base B [8]. The base consists of objects which are small and irreducible. In a prime field F_P, where we identify the field elements with integers in *0, 1,* · · · *, p − 1*, a factor base consists of all prime numbers less than some prescribed bound. In a field of characteristic 2, F_{2^n} , where we write field elements as polynomials of degree < *n*, a factor base consists of all irreducible polynomials of degree less than some prescribed bound.

The analysis of ICM is studied extensively. The efficient way of solving the DLP using the ICM was first presented by Coppersmith and Odlyzko [3]. Another variant of the ICM is the number sieve field and it has an heuristic running time of the form $\exp((c + o(1))(\log p)^{1/3} (\log\log p)^{2/3}$ [6, 14, 17, 18]. Introduction of the ICM on an elliptic curve group is a well known open problem [13]. Recently, the ICM was introduced in the DLP for hyper elliptic curves [2].

The special-purpose algorithms need some additional information apart from *g* and y

to solve the DLP. The attacks that are developed based on the information other than *g* and y are one way of solving the DLP through trap doors. One of the popular attacks of this kind, namely, the Pohlig-Hellman method [12] solves the DLP, when the factors of *p − 1* are small. This method reduces the DLP in a field to small subgroups. For example, if *p − 1* is a product of small factors, namely q_i, which are relatively prime to each other, then the method reduces the discrete logarithm $x \bmod p$ to $x_i \bmod q_i$, computes $x_i \bmod q_i$ in each q_i and finally combines the results using the Chinese Remainder Theorem. Similarly Van Oorschot and Wiener [16] presented the difficulty of computing the DLP, which is known to be short on a random prime with the factors of $p - 1$ consists of small factors along with one large factor (q).

In the present study, a variant of the ICM is proposed to solve the DLP for ephemeral keys. The problem is formulated as follows: Once the logarithms of a subset of group elements are known, the logarithms of the ephemeral keys can be solved by using the individual logarithm phase. Therefore, an efficient way of performing the individual logarithm step based on precomputation step is proposed. The newly proposed algorithms for the ICM are analyzed experimentally. The DLP of ephemeral keys is solved by using other methods, which are efficient on random primes, such as the Pohlig-Hellman method, the traditional individual logarithm step of the ICM and the Van Oorschot method. The results are compared with the newly proposed individual logarithm step of the ICM. Also, the DLP of ephemeral keys used in the Chang and Chang password key exchange protocol are solved. The rest of the paper is organized as follows: The following section presents the Pohlig- Hellman, the ICM and Van Oorschot methods. Section 2 discusses the variant of the ICM and the experimental analysis. Also, the other methods to solve the DLP for ephemeral keys are addressed. Section 3 reviews the Chang and Chang key exchange protocol. Section 4 presents the key recovery from the Chang and Chang password key exchange protocol and section 5 is the conclusion.

1.1 General algorithm for the Pohlig-Hellman and the Index Calculus Method

1.2 The Pohlig-Hellman Algorithm

This is an algorithm introduced by Pohlig-Hellman. If the order of the group is known along with the complete factorization and the factors are relatively small then this attack is possible.

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

$g^x = y \bmod p \Rightarrow$ can be reduced into
$\alpha^x = \beta \bmod p$ where
α is $g^{\left(\frac{p-1}{p_1^{e1}}\right)}, g^{\left(\frac{p-1}{p_2^{e2}}\right)}, \ldots g^{\left(\frac{p-1}{p_k^{ek}}\right)}$ and
β is $y^{\left(\frac{p-1}{p_1^{e1}}\right)}, y^{\left(\frac{p-1}{p_2^{e2}}\right)}, \ldots y^{\left(\frac{p-1}{p_k^{ek}}\right)}$

The logarithms 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 $x_i \bmod p_i^{e_i}$ to retrieve $x \bmod p$.

1.3 The Index Calculus Method

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 [8] is that if

$$\prod_{i-1}^{m} x_i = \prod_{j-1}^{n} y_j \qquad (1)$$

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

$$\sum_{i-1}^{m} \log_g x_i = \sum_{j-1}^{n} \log_g y_j (\bmod q-1) \qquad (2)$$

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.

The algorithm has two steps:

- A pre-computation step, where the logarithms of $\log_g b$ of all members of the factor base are obtained where g is the generator and b is the element in the factor base.
- A computation step, which tries enough $g^a y$ until the result factors over the factor base, thus providing the requested logarithm $\log_g y$, where y is the element for which the logarithm is to be computed.[15].

The pre-computation step itself has two phases

- First phase: Find the linear relations relating to the logarithms of the primes in the factor base
- Second phase: Solve this linear system using techniques from linear algebra

The general algorithm for the traditional ICM is described below [1].

1.3.1 General Algorithm for the ICM

INPUT a generator g of a cyclic group G of order n i.e., $p-1$ and an element y
OUTPUT $\log_g^y$

- pre-computation step
  - Select a factor base $S = \{p_1, p_2, ... p_t\}$, which belongs to G such that a significant portion of elements of G can be efficiently expressed as products of elements from S.
  - Find a linear system using the procedure as given below

  * Select a random integer k, such that $0 \le k \le n-1$ and compute g^k
  * Try to write g^k as a product of elements in S as

$$g^k = \prod_{i=1}^{t} p_i^{c_i}, c_i > 0 \qquad (3)$$

for any k. Then,

$k = \sum_{i=1}^{t} c_i \log_g p_i$

* Repeat the above steps to get the value of $t+c$ equations.

- The linear system is reduced into smaller size using a structured Gaussian method. This step is an optional one and is used when a large system is generated in the previous step.
- Solve this linear system to obtain $\log_g p_i$.

- Computation step
  - Compute $\log_g y$

* Select a random integer, $k,(0 \le k \le n-1)$ and compute yg^k.
* Try to write yg^k as a product of elements in S

$$yg^k = \prod_{i=1}^{t} p_i^{d_i} \tag{4}$$

for any k. Then,

$$\log_g y = \left(\sum_{i=1}^{t} d_i \log_g p_i - k\right) \bmod \mathrm{n}$$

1.4 The Van Oorschot method

Van Oorschot and Wiener [16] proposed an attack on short exponents with the combination of random prime. The algorithm is a combination of the Pohlig-Hellman and Pollard Lambda methods to solve the DLP with the above constraints and it works as follows: Let $y = g^x$ be an element of a group G of order $n = zQ$, where $z = B_r$ is the product of smooth factors, and has a bit length of approximately *k*. Compute V where $V = x \bmod z$, by a partial Pohlig-Hellman decomposition. Write $x = Az + V$, where $0 \le V < z$ with A as yet unknown. Then $y = g^x = g^{Az+V}$. Now $A \in [0,2^C]$, where $C = u - k$ bits of x remain unknown after finding V. Computing g^V and $y^* = y/g^V = g^{Az} = h^A$, where $h = g^z$ is known. Now V is to be computed from the lambda method. Since A and V are known, x can be calculated as $x = Az + V$.

2. METHODS TO SOLVE THE DLP FOR EPHEMERAL KEYS

In this section a variant of the ICM is proposed. The performance of newly proposed algorithms for the ICM are analyzed and reported. The steps involved in the newly proposed ICM are as follows:-

- A pre-computation step, where the logarithms of all subgroup elements are obtained.
- A computation step, which computes the logarithm of y by combining the logarithms of subgroup elements by using the Chinese Remainder Method.

2.1 Performance study and numerical results

In this section the experimental analysis on the new variant of the ICM is presented. The problem is described as follows: First a data file is produced, which contains a list of tuples. A tuple is of the form *(m, p, q, g, y)* with the following properties:- *m* lies between 13 and 50 digits, *p* is a prime, *q* is the list of factors of $p - 1$, *g* is the generator and *y* is an element of the prime field, such as $y = g^x$, with *x* to be recovered from *g* and *y*. Having built up the data file, the algorithms (pre-computation and individual logarithm step) for a variant of the ICM are implemented and verified on the data file. The selected list of problems and the running time to solve the DLP for *y* is reported in table 1. The DLP of ephemeral keys can be solved efficiently, once the logarithms of a subset of group are known. Assume the logarithms of a subset of a group i.e., the logarithms of subgroup elements, are computed by using *Algorithm-1.* Since the logarithms of subgroup elements are known, the DLP for the ephemeral keys can be obtained by using *Algorithm-2*. This is possible due to the fact that the prime field and the generator are shared between the communicators before starting the sessions. The assumption is that the logarithms of subgroup elements are computed before starting the sessions. The DLP for ephemeral keys is to be computed once the session gets started.

Apart from the proposed individual logarithm step of the ICM, the other methods, such as the traditional individual logarithm phase of the ICM, the conventional Pohlig-Hellman and Van Oorschot methods are used to solve the DLP for ephemeral keys. Since, the pre-computation step is related to the traditional ICM and a variant of the ICM proposed in the present study, the Pohlig-Hellman and the Van Oorschot methods are implemented without considering the pre-computation step. The pre-computation step of the traditional individual logarithm step is the conventional pre-computation step of the ICM as mentioned in section 1.3.1. Finally, the exponents (x) are assumed as short for the Van Oorschot method.

Algorithm-1 To find the logarithm of all subgroup elements in the order of subgroups.
INPUT Problem of size *p* and factors of *p-1*.
OUTPUT Logarithm of all small subgroup elements in the order of subgroups.
Known information is *g* : generator of Z^*_p , and factors of $p - 1$.

```
for every subgroup of order $P_i$ do
    Assign $G = g^{P-1/P_i}$
    Assign $A = P_i$
    for $i$ in $1..A$ do
        Assign $H = G^i$
        Assign $\log h = i * p - 1/ p_i$
        Store them in the list
    end for
end for
```

Algorithm-2 To find the logarithm of ephemeral key
INPUT Problem of size *p* and factors of $p - 1$.
OUTPUT Logarithm of ephemeral key.
Known information is *g* : generator of $Z^*_P , y :$ element of Z^*_P and factors of $p - 1$.

```
for every factor $p_i$ of $p - 1$ do
    Assign $A_i$ as $y^{p-1/p1}$
    Obtain the logarithm of $A_i$ from the pre-computed list
end for
Combine the results $A_i$ mod $p_i$ by using the Chinese Remainder Method to obtain $y$ mod $p - 1$
The DLP of y (the ephemeral key) is obtained
```

Table 1. Running time of a variant of the ICM

Problem	Running time in secs.
4324122104434447665362908248086967822904859	55
40500691568928903388503314943591776516203	88
10548813247704246266317485054480132114947	31
22750475822981512251147389834477659827887	29
31023376122247516706110077047014990674391	26
164838881183106336031117884004666831843	6
29565696133579269116146450939411987039	3
175660822294031990216013131647599098066411	203
116045117879379642828484543996278641093	181
96834800300461810039999000830418556963	651
40687382369048479475232114904989637283	633
247747816765350307022689158418187059	171
38174514779333277109099448511590627	161

Table 2. Running time to solve the DLP for ephemeral keys

Problem size in digits	Van Oorschot method	Individual logarithm phase of the ICM	Traditional individual logarithm phase of the ICM	Pohlig-Hellman method
19	8 ms	6 ms	104 ms	22 ms
20	5 ms	10 ms	300 ms	31 ms
21	5 ms	173 ms	407 ms	103 ms
22	9 ms	18 ms	605 ms	42 ms
23	5 ms	8 ms	3s	54 ms
24	4 ms	4 ms	4s	13 ms
25	6 ms	16 ms	7s	54 ms
26	6 ms	4 ms	37s	14 ms
27	5 ms	400 ms	61s	51 ms
28	10ms	5 ms	194s	16 ms

The running time of the newly proposed individual logarithm step of the ICM is compared with the other methods. Table 2 reports the running time of the above methods. Since the exponents (x) are assumed as short, the Van Oorschot method solves the problem in reduced time. The traditional individual logarithm step needs substantially more time due to the fact that the DLP is solved without considering the additional information such as the factors of $p - 1$. The Pohlig-Hellman method and the variant of the ICM work with the additional information regarding the factors of $p - 1$ and their running time depends on the size of the factors of $p - 1$. The following section discusses the ephemeral key recovery on the Chang and Chang password key exchange protocol.

3. Review of the Chang and Chang Novel Three Party Key Exchange Protocol

The key exchange protocol is one of the most elegant ways of establishing secure communication between a pair of users by using a session key. The session key, which is exchanged between two users assures secure communication for later sessions. The first practical key exchange protocol was proposed by Diffie-Hellman [5]. Since the introduction of key exchange protocol by Diffie-Hellman, various versions and improvements in key exchange protocol have been developed. Recently Chang and Chang [4] proposed a novel three party encrypted key exchange protocol and claimed the protocol is secure, efficient and practical.

This section briefly explains the Chang and Chang novel three party key exchange protocol. The notations used in this protocol are listed below:

- *A,B* : two communication parties.
- *S*: the trusted server.
- ID_A, ID_B, ID_S : the identities of A,B and S, respectively.
- PW_A, PW_B : the passwords securely shared by A with S and B.
- $E_{PW}(.)$: a symmetric encryption scheme with a password PW.
- r_A, r_B : the random numbers chosen by A and B, respectively.
- p : a large prime.
- g : a generator of order $p-1$.
- R_A, R_B, R_S : the random exponents chosen by A,B and S, respectively.
- $N_A, N_B : N_A = g^{R_A} \bmod P, N_B = g^{R_B} \bmod p$.
- $F_S(.)$: the one-way trapdoor hash function (TDF) where only S knows the trapdoor.
- $f_K(.)$: the pseudo-random hash function (PRF) indexed by a key K.
- K_{AS}, K_{BS} : one time strong keys shared by A with S and B with S, respectively.

The procedure followed in Chang-Chang is given below:

Step 1:

A → B : $\{ID_A, ID_B, ID_S, E_{PW_A}(N_A), F_S(r_A), f_{KAS}(N_A)\}$ User A chooses a random integer number r_A and a random exponent $R_A \in_R Z_P$, and then computes $N_A = g^{R_A} \bmod P$ and $K_{AS} = N^{R_A}$. Then, A encrypts N_A by using his/her password PW_A like $E_{PW_A}(N_A)$ and computes two hash values $F_S(r_A), f_{KAS}(N_A)$. Finally, A sends $\{ID_A, ID_B, ID_S, E_{PW_A}(N_A), F_S(r_A), f_{KAS}(N_A)\}$ to B.

Step2:

B → S: $\{ID_A, ID_B, ID_S, E_{PW_A}(N_A), F_S(r_A), f_{KAS}(N_A), E_{PW_B}(N_B), F_S(r_B), f_{KBS}(N_B)\}$ User B chooses a random integer r_B and a random

Exponent $R_B \in_R Z_P$, and then computes $N_B = g^{R_B} \bmod p$ and $K_{BS} = N^{R_B}$. Then, B encrypts N_B by using his/her password PW_B like $E_{PWB}(N_B)$ and computes two hash values $F_S(r_B), f_{KBS}(N_B)$. Finally,B sends $\{ID_A, ID_B, ID_S, E_{PW_A}(N_A), F_S(r_A), f_{KAS}(N_A), E_{PW_B}(N_B), F_S(r_B), f_{KBS}(N_B)\}$ to S.

Step3:

S → B: $\{N_B{}^{R_S}, f_{KAS}(ID_A, ID_B, K_{AS}, N_B{}^{R_S}), N_A{}^{R_S}, f_{KBS}(ID_A, ID_B, K_{BS}, N_A{}^{R_S})\}$, Server S decrypts $E_{PW_A}(N_A)$ and $E_{PWB}(N_B)$ by using PW_A, PW_B to get N_A, N_B respectively. Then, S gets r_A and r_B from $F_S(r_A)$ and $F_S(r_B)$ by using a trap door, respectively. To authenticate A and B, S computes $K_{AS} = N^{R_A}$ and $K_{BS} = N^{R_B}$ and then verifies $f_{KAS}(N_A)$ and $f_{KBS}(N_B)$, respectively. If successful, S chooses a random exponent $R_S \in_R Z_P$ and then computes $N_A{}^{R_S}$ and $N_B{}^{R_S}$, respectively. Finally, S computes two hash values $f_{KAS}(ID_A, ID_B, K_{AS}, N_B{}^{R_S}), f_{KBS}(ID_A, ID_B, K_{BS}, N_A{}^{R_S})$ to B.

Step 4:

B → A: $N_A^{R_S} f_{KAS}(ID_A, ID_B, K_{AS}, N_B^{R_S}), f_K(ID_A, K)\}$ .By using $K_{BS} = N^{R_B}$, B authenticates S by checking $f_{KBS}(ID_A, ID_B, K_{BS}, N_A^{R_S})$.

If successful, B computes the session key $K = (N_A^{R_S})^{R_B} = g^{R_A R_B R_S}$ and hash value $f_K(ID_B, K)$, and then sends $\{N_B^{R_S}, f_{KAS}(ID_A, ID_B, K_{AS}, N_B^{R_S}), f_K(ID_B, K)\}$ to A.

Step5:

A → B: $f_K(ID_A, K)$ By using $K_{AS} = N^{R_A}$, A authenticates S by checking $f_{KAS}(ID_A, ID_B, K_{AS}, N_B^{R_S})$. If successful A computes the session key $K = (N_A^{R_S})^{R_A} = g^{R_A R_B R_S}$, and authenticates B by checking $f_K(ID_B, K)$. If authenticates is passed, A computes and sends $f_K(ID_A, K)$

Step 6:

B authenticates A by checking $f_K(ID_A, K)$. If successful, B confirms A's knowledge of the session key $K = g^{R_A R_B R_S}$.

4. A KEY RECOVERY ATTACK ON THE CHANG AND CHANG PROTOCOL

In this section the key recovery attack on the Chang and Chang protocol proposed by R.Padmavathy and Chakravarthy Bhagvati is reviewed [10]. A malicious party B guesses the password of A using an undetectable password guessing attack as proposed by Yoon and Yoo [19]. B uses the password of A for obtaining the session key between A and C, when A and C want to communicate. The following procedure presents the attack in detail.

Step 1:

A → C : $\{ID_A, ID_B, ID_S, E_{PW_A}(N_A), F_S(r_A), f_{KAS}(N_A)\}$ User A chooses a random integer number r_A and a random exponent $R_A \in_R Z_P$, and then computes $N_A = g^{R_A} \bmod P$ and $K_{AS} = N^{R_A}$. Then, A encrypts N_A by using his/her password PW_A like $E_{PW_A}(N_A)$ and computes two hash values $F_S(r_A), f_{KAS}(N_A)$ Finally, A sends $\{ID_A, ID_B, ID_S, E_{PW_A}(N_A), F_S(r_A), f_{KAS}(N_A)\}$ to C.

Step2:

B gets $\{ID_A, ID_B, ID_S, E_{PW_A}(N_A), F_S(r_A), f_{KAS}(N_A)\}$ and from $E_{PW_A}(N_A)$ decrypts N_A since the password is known and solves the ephemeral key R_A from N_A using the popular Pohlig-Hellman method or index calculus method as discussed above. This is achieved, since the order of the generator used in the protocol is $p-1$. When the factorization of $p-1$ is known and small, the exponent can be solved in reduced time, even in polynomial time when the factors of $p-1$ is in the form of $2^n p_1^m$, where p_1 is small and n, m are very large.

Step3:

C → S: $\{ID_A, ID_C, ID_S, E_{PW_A}(N_A), F_S(r_A), f_{KAS}(N_A), E_{PW_C}(N_C), F_S(r_C), f_{KCS}(N_C)\}$ User C chooses a random integer r_C and a random exponent $R_C \in_R Z_P$, and then computes $N_C = g^{R_C} \bmod p$ and $K_{CS} = N^{R_C}$. Then, C encrypts N_C by using his/her password PW_C like $E_{PWC}(N_C)$ and computes two hash values $F_S(r_C), f_{KCS}(N_C)$. Finally, C sends $\{ID_A, ID_C, ID_S, E_{PW_A}(N_A), F_S(r_A), f_{KAS}(N_A), E_{PW_C}(N_C), F_S(r_C), f_{KCS}(N_C)\}$ S.

Step4:

S → C: $\{N_C^{R_S}, f_{KAS}(ID_A, ID_C, K_{AS}, N_C^{R_S}), N_A^{R_S}, f_{KCS}(ID_A, ID_C, K_{CS}, N_A^{R_S})\}$ Server S decrypts $E_{PW_A}(N_A)$ and $E_{PW_C}(N_C)$ by using PW_A, PW_C to get N_A, N_C, respectively. Then, S gets r_A and r_C from $F_S(r_A)$and $F_S(r_C)$ by using a trap door, respectively. To authenticate A and B, S computes $K_{AS} = N^{r_A}$ and $K_{CS} = N_C^{r_C}$ and then verifies $f_{KAS}(N_A)$ and $f_{KCS}(N_C)$, respectively. If successful, S chooses a random exponent $R_S \in_R Z_P$ and then computes, $N_A^{R_S}$ and, $N_C^{R_S}$, respectively. Finally, S computes two hash values $f_{KAS}(ID_A, ID_B, , K_{AS}, N_B^{R_S}), f_{KCS}(ID_A, ID_C, K_{CS}, N_A^{R_S})$ and sends $\{N_C^{R_S} f_{KAS}(ID_A, ID_C, K_{AS}, N_C^{R_S}), N_A^{R_S}$, $f_{KCS}(ID_A, ID_C, K_{CS}, N_A^{R_S})\}$ to B.

Step5:

B gets $\{N_C^{R_S} f_{KAS}(ID_A, ID_C, K_{AS}, N_C^{R_S}), N_A^{R_S}, f_{KCS}(ID_A, ID_C, K_{CS}, N_A^{R_S})\}$ and from $N_C^{R_S}$ he computes the session key $K = (N_C^{R_S})^{R_B}$ is nothing but a session key between A and C $g^{R_A R_B R_S}$

Step6:

C → A: $\{N_C^{R_S} f_{KAS}(ID_A, ID_C, K_{AS}, N_C^{R_S}), f_K(ID_C, K)\}$ By using $K_{CS} = N_C^{r_C}$, C authenticates S by checking $f_{KCS}(ID_A, ID_C, K_{CS}, N_A^{R_S})$. If successful, C computes the session key $K = (N_A^{R_S})^{R_B} = g^{R_A R_B R_S}$ and hash value $f_K(ID_C, K)$ and then sends $\{N_C^{R_S} f_{KAS}(ID_A, ID_C, K_{AS}, N_C^{R_S}), f_K(ID_C, K)\}$ to A.

Step7:

A → C: $f_K(ID_A, K)$ By using $K_{AS} = N^{r_A}$, A authenticates S by checking $f_{KAS}(ID_A, ID_C, K_{AS}, N_C^{R_S})$. If successful A computes the session key $K = (N_A^{R_S})^{R_A} = g^{R_A R_B R_S}$, and authenticates C by checking $f_K(ID_C, K)$. If authenticates is passed, A computes and sends $f_K(ID_A, K)$

Step8:

C authenticates A by checking $f_K(ID_A, K)$. If successful, C confirms A's knowledge of the session key $K = g^{R_A R_B R_S}$.

4.1 Solving the DLP for the ephemeral keys NA and NB

From the discussion of the key recovery attack on the Chang-Chang protocol, it is observed that the computation of R_A from N_A is the key issue. This leads to recovery of the session key by the malicious party B. In this section the method of computing the key R_A form N_A by solving the Discrete Logarithm Problem (DLP) is presented. The problem is described as follows: First a data file is produced, which contains a list of tuples. A tuple is of the form *(m, p, q)* with the following properties. *m* lies between 13 and 200 digits, *p* is a prime, *q* is the list of factors of $p - 1$. Having built up the data file the Chang-Chang password key exchange protocol is implemented and the DLP of the ephemeral keys (N_A and N_B) are solved by using the methods discussed in sections 1 and 2. The following table shows the selected list of problems and the running time to recover the session key using key recovery attack. On a similar line the table 4 tabulates the selected list of problems and the running time to compute N_A and N_B from the Chang and Chang password key exchange protocol.

Table 3. Running time to recover the session key

problem	Running time to recover the session key
4324122104434447665362908248086967822904859	9ms
40500691568928903388503314943591776516203	14 ms
10548813247704246266317485054480132114947	8 ms
22750475822981512251147389834477659827887	8 ms
31023376122247516706110077047014990674391	9 ms
164838881183106336031117884004666831843	5 ms
29565696133579269116146450939411987039	5 ms

Table 4. Running time to solve N_A or N_B

problem	Running time to solve N_A and N_B
1910009901593650473786381403548828023839870277948686259673707683	10ms
426927464819150521468225593921054541511202645719071417269428709	12ms
35305968189548375940583635702770605868345200000000000000000000000000000000000000000000000000002298948118662442499371103438785908001117297699	14h10m3s
2455056898110742462098979121323214291116549232201635656436531252489977023983947006356459131339580436008715638751116 8667088725749006168938241077	16h5m2s
381433650162875350269924530881206967174557955208354469402441808891292308580580746717399	12m11s
083544694024418088912923085805807467173997648396681212247460911159847224901457171	14m11s

For example, a problem of given p = 1235413821472326940353348521633205352466011099496309478524737854504860659098018702216840863724559053405840009776995133914517655221962730411575270239409171310618340555720351403579115867614746093751 is solved in a fraction of seconds.

5. Conclusion

This paper discussed a new approach to solve the DLP for ephemeral keys. A new variant of the ICM is presented and the performance is analyzed using a comprehensive set of experiments. The Pohlig-Hellman method is the best known method when the factors of $p - 1$ are small. The ICM is an efficient method for the general DLP. Through the experimental results it is shown that the individual logarithm step of the ICM outperforms the Pohlig-Hellman method for most of the cases. The other methods, which are efficient on random primes, such as the traditional ICM and the Van Oorschot method are also analyzed and compared with the newly proposed method for the ICM. It is also shown that the ephemeral keys of the Chang and Chang password key exchange protocols are solved by using the above methods to recover the session key exchanged between two communicators. The attack can be avoided by selecting the generators of large prime order and the computations are to be restricted to the prime order subgroup of $p - 1$.

REFERENCES

[1] J. Buchmann and D. Weber, "Discrete Logarithms:Recent Progress," *Technical report*, no:T1-12/98.

[2] H. Cohen, and G. Fery, '*Handbook of Elliptic and Hyperelliptic Curve Cryptography*,' Discrete Mathematics and Applications, CRC Press, 2005.

[3] D. Coppersmith, A. M Odlyzko, and R. Schroeppel, "Discrete logarithms in GF(p)," *Algorithmica*, v1, pp.1-15, 1986.

[4] CC. Chang., YF. Chang, "A novel three party encrypted key exchange protocol," *Computer Standards and Interfaces*, v26(5), pp.471-6, 2004.

[5] W. Diffie, and M. Hellman, "New Directions in cryptography," *IEEE Transaction on Information The ory*, v22(6), pp.644-54, 1976.

[6] D. M Gordon, "Discrete logarithms in GF(p) using the number field sieve," *SIAM Journal of Discrete Mathematics*, v6, pp.124-138, 1992.

[7] D. E Knuth, *The Art of computer programming,vol.3:Sorting and Searching*, Addison-Wesley, 1973.

[8] McCurely, "The Discrete logarithm problem," *Cryptology and computational number theory proceeding of symposia in Applied Mathematics*, v42, pp.49-74.

[9] A. Menezes, and U. Berkant, On Reusing Ephemeral Keys in Diffie-Hellman Key Agreement Protocols, preprint, 2008.

[10] R. Padmavathy, and Chakravarthy Bhagvati, "A Key Recovery Attack on Chang and Chang Password Key Exchange Protocol," *International Conference on Computer and Network Technology*, 2009.

[11] J. M Pollard, "Monte Carlo methods for index computation (mod p)," *Mathematics of Computation.*, v32(143), pp.106-110, 1978.

[12] S. Pohlig, and M. Hellman, "An improved algorithm for computing logarithms over GF(p) and its cryptographic significance," *IEEE Transaction on Information Theory*, v24, pp.106-110, 1978.

[13] J. Silverman, "The xedni calculus and the elliptic curve logarithm problem," *Design Codes and Cryptography*, v20, pp.5-40, 2000.

[14] O. Schirokauer, D.Weber and T. Denny, "Discrete logarithms the effectiveness of the index calculus method," *Proceeding of ANTS II, LNCS v1122*, pp.337-361, 1996.

[15] C. Studholme, Discrete logarithm problem, *Research paper requirement (milestone) of the PhD program at the University of Toronto*, June 21, 2002.

[16] P. C, Van Oorschot and M. J, Wiener, "On Diffie-Hellman Key agreement with short Exponents," Proceeding of *Eurocrypt* LNCS v1070, pp.332-343, 1996

[17] D. Weber, "Computing Discrete logarithms with the general number field sieve," *Proceeding of ANTS II*, LNCS v1122, pp.99-114, 1996.

[18] D.Weber and T. Denny, The solution of McCurleys discrete log challenge, *Proceeding of Crypto98*, LNCS v1462, pp.458-471, 1998.

[19] EJ. Yoon and KY. Yoo, "Improving the novel three-party encrypted key exchange protocol," *Computer Standards and Interfaces*, v30, pp.309-314, 2008.

R. Padmavathy

She received an M.tech degree from Andhra University and a Ph.D from the University of Hyderabad, India. At present she is working as a faculty member at the National Institute of Technology, Warangal. Her research interests include Information security, Cryptology and Network Security.

Chakravarthy Bhagvati

He received his Ph.D degree from RPI Newyork, USA . At present he is a professor at University of Hyderabad, Hyderabad, India. He published a number of papers in International Conferences and Journals. Recently he chaired the National Workshop on Cryptology organized by the CRSI-Cryptology Research Society of India and an International Workshop on Computer Vision organized by IIIT-Hyderabad. His research interests include Image Processing, Computer Vision, Pattern Recognition, OCR for telugu and Cryptography.