Efficient algorithm to solve DLP from partial knowledge of the Key

N.Anil Kumar
C R Rao Advanced Institute of Mathematics, Statistics and Computer Science (AIMSCS), Prof C R Rao road, Hyderabad-500 046, INDIA.
e-mail: anil230@gmail.com, anil@crraoaimscs.res.in

Chakravarthy Bhagvati
Department of Computer and Information Sciences, University of Hyderabad, Prof C R Rao raod, Hyderabad-500 046, INDIA.
e-mail: chakcs@uohyd.ernet.in

Abstract—**The discrete logarithm problem (DLP) is assumed to be hard from computational point of view. Side channel attacks are used to reveal information about the key if proper countermeasures are not used. We study the discrete logarithm problem assuming that partial information about the key is known. K.Gopalakrishnan et al. provided algorithms to solve the discrete logarithm problem for generic groups which are considerably better than square-root attack on the whole key when one contiguous bits of the key is revealed. This paper provides algorithms which will work when more than one contiguous bits of the key are revealed with time complexity of order square root $2^{(unknownbits)}$**

Keywords-Discrete Logartihm Problem; partial knowledge of key; Elliptic curves; Pollard's kangaroo Algorithm;

I. INTRODUCTION

The discrete logarithm problem (DLP) is assumed to be hard from computational point of view in modern cryptography. Side channel attacks are used to reveal information about the key if proper countermeasures are not used. We assume that a partial knowledge of the key is known either by side channel analysis [6] or through other ways and try to solve the DLP. We try to solve DLP when one or more contiguous bits of the key are revealed. From the partial knowledge we try to solve the discrete logarithm assuming only the group structure

II. BACKGROUND

Let G be a cyclic group. Let $g \in G$ of order Ord(g). Given β an element in the sub group generated by g determine $\alpha \in \{1,2,\ldots,Ord(g)\}$ such that $\alpha\, g=\beta$. Here β and g are public parameters and the secret key is α, finding the secret key α from the public parameters β and g is called Discrete Log Problem (DLP). Although the description is given for additive groups the description for multiplicative groups is almost similar.

III. SOLVING DLP FROM PARTIAL KNOWLEDGE OF KEY

Gopalakrishnan et al [4] has presented an algorithm that improves on the exhaustive search when the partial information of the key is revealed. We will briefly describe their method. Here we assume that a sequence of contiguous bits of α is revealed with the position information. This can be divided into three cases

1. Left part is revealed.
2. Right part is revealed.
3. Middle part is revealed.

A. *Left part is revealed*

We assume that contiguous most significant bits of the key are known. Let z denotes the bit string formed by the sequence of known bits. Then the smallest possible value for α is the number a represented by z concatenated with a sequence of zeros (up to max digits). The largest possible value for α is the number b represented by z concatenated with a sequence of ones (up to max digits). We know $a \le \alpha \le b$. We can use Pollard's Kangaroo Algorithm[7] with expected running time of $O(\sqrt{b-a})$

B. *Right part is revealed*

We assume that contiguous least significant bits of the key are known. So we can write $\alpha = \alpha_1 \times 2^{l_2} + \alpha_2$ where α_2 is known and α_1 is unknown.

Let $M = \left\lfloor \dfrac{p-\alpha_1 - 1}{2^{l_2}} \right\rfloor$, then we know that $0 \le \alpha_1 \le M$ since $0 \le \alpha \le p-1$.

$$\beta = \alpha g = (\alpha_1 \times 2^{l_2} + \alpha_2) g = (g \times 2^{l_2})\alpha_1 + g \times \alpha_2$$

If we denote $2^{l_2} g$ by g' and $\beta - \alpha_2 g$ by β' the above equation reduces to $\beta' = \alpha_1 g'$. We can then solve this DLP by using Pollard's Kangaroo Algorithm on g' and β' in $O(\sqrt{M})$

C. *Middle part is revealed*

Let α be the key. We have positive integers M and N such that $\alpha = \alpha_1 MN + \alpha_2 M + \alpha_3$ Where α_1 and α_3 are unknown and α_2 is known [4].

Let p (prime) is the order of g. Let r be an integer $0 < r < p$ such that $r\,M\,N$ as $k\,p + s$ with $|s| < p/2$

$$r\alpha = r\alpha_1 MN + r\alpha_2 M + r\alpha_3$$
$$= \alpha_1 kp + s\alpha_1 + r\alpha_2 M + r\alpha_3$$
$$= \alpha_1 kp + r\alpha_2 M + \alpha'$$

Where $\alpha' = s\alpha_1 + r\alpha_3$. Multiplying g to both sides we get

$$g \times r\alpha = g \times (\alpha_1 kp + r\alpha_2 M + \alpha')$$
$$(g\alpha)r = (gp)\alpha_1 k + gr\alpha_2 M + g\alpha'$$
$$\beta r = gr\alpha_2 M + g\alpha'$$
$$\beta' = g\alpha' \text{ where } \beta' = (\beta - g\alpha_2 M)r$$

The interval of α' is $\left[0, r(M-1) + s\left(\frac{p}{MN} - 1\right)\right]$. The time complexity is $O(\sqrt{2p} / N^{\frac{1}{4}})$

After computing $\alpha' = (\alpha_3 r + \alpha_1 s)$ we have to extract α_1 and α_3. We can assume r and s to be co-prime (if they are not co- prime we can take the gcd of r and s and divide the whole equation). Well known number theoretic techniques give us that all solutions are of the form

$$\alpha_1 = b + ir$$
$$\alpha_3 = \frac{\alpha' - sb}{r} - is$$

Where $b \equiv \alpha' s^{-1} \bmod r.$ As we know $|s| < p/2$ which implies $r > \frac{p}{2MN}$ and $0 \le \alpha_1 < \frac{p}{MN}$, there are at most two possibilities which can be easily determined.

IV. OUR OBSERVATION ON GOPALAKRISHNAN ET AL[4] ALGORITHM

The method suggested by Gopalakrishnan et al [4] to extract α_3 and α_1 from α' may not give the solution as it does not take into account all possible solutions for α_3 and α_1. So we suggest the following procedure.

As α' is an element of Z_p the exact value of α' can be $\alpha' + jp$ where j can take integer values like 0,1,2,…; We have to observe that the equation $\alpha' \equiv (\alpha_3 r + \alpha_1 s) \bmod p,$ can be written as $\alpha' + jp = (\alpha_3 r + \alpha_1 s).$

Now the parametric equations can be written as $\alpha_1 = b + ir$ $\alpha_3 = \frac{\alpha' - sb}{r} - is$ Where $b \equiv \alpha' s^{-1} + jp \bmod r.$ For different values of i and j we will get different α_1 and α_3. For each value we can construct α by (as α_2 is known)

$$\alpha = \alpha_1 MN + \alpha_2 M + \alpha_3$$

For each value we will check whether $\alpha g = \beta$ or not, if for some value of α the equation satisfies, that is the required key.

V. PROPOSED METHOD

In this section we propose a method that works when two or more contiguous bits of the key are know. First describe when two contiguous bits are reveled and later on generalize it.

We assume that two contiguous bits of the key α are revealed. So the key is divided into five parts. Here $M_5 = 1$ if we take α_5 as least significant bits. We assume that α_2, α_4 are known and $\alpha_1, \alpha_2, \alpha_3$ are unknown. M_1, M_2, M_3, M_4 and M_5 are positive integers such that the following equation holds. (Usually M_1, M_2, .. are in the form $2^{l_1}, 2^{l_2}, \ldots$).

$$\alpha = \alpha_1 M_1 + \alpha_2 M_2 + \alpha_3 M_3 + \alpha_4 M_4 + \alpha_5 M_5$$

We know α_2 and α_4 substituting in the equation $\alpha g = \beta$ we get

$$\alpha_1 M_1 g + \alpha_3 M_3 g + \alpha_5 M_5 g = \beta_1 \text{ where}$$
$$\beta_1 = \beta - \alpha_2 M_2 g - \alpha_4 M_4 g$$

A. *Solution by equation*

Substituting different values for α_1, α_3, α_5 as we know the number of the bits of α_1, α_3, α_5 we generate random values like $\alpha_{11}, \alpha_{31}, \alpha_{51}$ and a_1 whose bit length is less than or equal to the max bit length of α_1, α_3, α_5. We get a set of equations like

$$\alpha_{1i} M_1 g + \alpha_{3i} M_3 g + \alpha_{5i} M_5 g - a_i \beta_1 = R_i \quad \ldots\ldots.. (1)$$

We will substitute the generated random values and get different R_i. If we get two equations with same R_i we can solve them for α.

Lemma 1 There exists some value of α_{1j}, α_{3j}, α_{5j} and a_j for which LHS of Eqn 1 is O

Proof: We know $\alpha g = \beta$ and $\alpha = \alpha_1 M_1 + \alpha_2 M_2 + \alpha_3 M_3 + \alpha_4 M_4 + \alpha_5 M_5$

So $(\alpha_1 M_1 + \alpha_2 M_2 + \alpha_3 M_3 + \alpha_4 M_4 + \alpha_5 M_5) g = \beta$

$$\alpha_1 M_1 g + \alpha_3 M_3 g + \alpha_5 M_5 g = \beta - \alpha_2 M_2 g - \alpha_4 M_4 P$$
$$\alpha_1 M_1 g + \alpha_3 M_3 g + \alpha_5 M_5 g = \beta_1$$

So there exists α_1, α_3 and α_5 such that LHS of Eqn 1 is O

Lemma 2: There exist non-trivial values α_{1j}, α_{3j}, α_{5j} and α_{1k}, α_{3k}, α_{5k} such that

$$\alpha_{1j}M_1g+\alpha_{3j}M_3g+\alpha_{5j}M_5g+a_j\beta_1=$$
$$\alpha_{1k}M_1g+\alpha_{3k}M_3g+\alpha_{5k}M_5g+a_k\beta_1$$

for some j and k.

Proof: As α_{11}, α_{12}, …, α_{31}, α_{32}, …, α_{51}, α_{52}, … are randomly generated we can assume without loss of generality for some j

$$\alpha_{1j}>\alpha_1,\ \alpha_{3j}>\alpha_3,\ \alpha_{5j}>\alpha_5$$

So let $\alpha_{1k}=\alpha_{1j}-\alpha_1$, $\alpha_{3k}=\alpha_{3j}-\alpha_3$, $\alpha_{5k}=\alpha_{5j}-\alpha_5$

Let $a_k=2$

$$\begin{aligned}
&\alpha_{1k}M_1g+\alpha_{3k}M_3g+\alpha_{5k}M_5g+a_k\beta_1\\
&\quad=\alpha_{1j}M_1g-\alpha_1M_1g+\alpha_{3j}M_3g-\alpha_3M_3g\\
&\qquad+\alpha_{5j}M_5g-\alpha_5M_5g+a_k\beta_1\\
&\quad=\alpha_{1j}M_1g+\alpha_{3j}M_3g+\alpha_{5j}M_5g+2\beta_1\\
&\qquad-(\alpha_1M_1g+\alpha_3M_3g+\alpha_5M_5g)\\
&\quad=\alpha_{1j}M_1g+\alpha_{3j}M_3g+\alpha_{5j}M_5g+2\beta_1-\beta_1\\
&\quad=\alpha_{1j}M_1g+\alpha_{3j}M_3g+\alpha_{5j}M_5g+\beta_1
\end{aligned}$$

Hence there exist nontrival values. So the lemma is proved. We can define a recurrence relation so that it can collide at non trivial values

1) Recurrence Relation: To keep the memory requirements at low we have to use Floyds cycle detection algorithm. Checking i and $2i$ in a sequence is sufficient to find collusion as proved propositon 11.1 in [5]. So we define it as a recurrence relation.

Let $r_1(0)$, $r_3(0)$, $r_5(0)$ and $r'(0)$ be choosen small integers or all zeros.

Let $E_0=r_1(0)M_1g+r_3(0)M_3g+r_5(0)M_5g-r'(0)\beta_1$

Let $t_i=$ x-coordinate of (E_i) and $b(\alpha_i)=$ no of bits in α_i

$l_u=$ Number of unknown bits, ord(g) is the order of point g.

$$r_1(i)=\begin{cases}r_1(i-1)+1 & \text{if } t_i<l(\alpha_1)\times p/l_u\\ r_1(i-1) & \text{otherwise}\end{cases}$$

$$r_3(i)=\begin{cases}r_3(i-1)+1 & \text{if } b(\alpha_1)\times p/(l_u+1)<t_i\le\\ & [b(\alpha_1)+b(\alpha_3)]\times p/l_u\\ r_3(i-1) & \text{otherwise}\end{cases}$$

$$r_5(i)=\begin{cases}r_5(i-1)+1 & \text{if } [b(\alpha_1)+b(\alpha_3)]\times p/(l_u+1)\\ & <t_i\le\\ & [b(\alpha_1)+b(\alpha_3)+l(\alpha_5)]\times p/l_u\\ r_5(i-1) & \text{otherwise}\end{cases}$$

$$r'(i)=\begin{cases}r'(i-1)+1 & \text{if } t_i\equiv 0 \bmod 3\\ r'(i-1) & \text{otherwise}\end{cases}$$

$$E_i=r_i(t_i)M_1g+r_3(t_i)M_3g+r_5(t_i)M_5g-r'(t_i)\beta_1$$

If E_i equals E_{2i} for some value of i then we get a relation which involves g and β_1 (which can be written in terms of β). As we know the order of g we can simplify for α.

$$\begin{aligned}
&r_1(t_i)M_ig+r_3(t_i)M_3g+r_5(t_i)M_5g-r'(t_i)\beta_1\\
&\equiv r_1(t_{2i})M_1g+r_3(t_{2i})M_3g+r_5(t_{2i})M_5g-r'(t_{2i})\beta_1\\
&\hfill \bmod ord(g)
\end{aligned}$$

$$\begin{aligned}
&[r_1(t_i)M_1+r_3(t_i)M_3+r_5(t_i)M_5-r_1(t_{2i})M_1-r_3(t_{2i})M_3\\
&\quad-r_5(t_{2i})M_5]g\equiv[r'(t_1)-r'(t_{2i})]\beta_1 \bmod ord(g)
\end{aligned}$$

Let $k_1=r_1(t_i)M_1+r_3(t_i)M_3+r_5(t_i)M_5-r_1(t_{2i})M_1$
$\quad -r_3(t_{2i})M_3-r_5(t_{2i})M_5$

Substituting the value of β_1

$$k_1g\equiv[r'(t_i)-r'(t_{2i})](\beta-\alpha_2M_2g-\alpha_4M_4g)$$
$$\bmod ord(g)$$

Rearranging gives

$$k_1g+(\alpha_2M_2+\alpha_4M_4)(r'(t_i)-r'(t_{2i}))g\equiv$$
$$[r'(t_i)-r'(t_{2i})]\beta \bmod ord(g)$$

Let $k_2=k_1+(\alpha_2M_2+\alpha_4M_4)(r_6(t_i)-r_6(t_{2i}))$ and $k_3=r'(t_i)-r'(t_{2i})$ then the equation becomes

$$k_2g=k_3\beta \bmod ord(g)$$

So $k_2k_3^{-1}g=\beta \bmod ord(g)$ if k_3 is invertible. So $k_3^{-1}k_2 \bmod ord(\alpha)$ is the required α

Similarly we can solve for the case when $E_i=-E_{2i}$

Let the number of unknown bits be l_u. The possible number of values the above equation can take is 2^{l_u}. We will get a collision in $O(\sqrt{2^{l_u}})$.

If we know three contiguous bits of the key then we can write

$$\alpha = \alpha_1 M_1 + \alpha_2 M_2 + \alpha_3 M_3 + \alpha_4 M_4 + \alpha_5 M_5 + \alpha_6 M_6 + \alpha_7 M_7$$

we know α_2, α_4, α_6 substituting in the equation $\alpha\ g = \beta$. we get $\alpha_1 M_1 g + \alpha_3 M_3 g + \alpha_5 M_5 g + \alpha_7 M_7 g = \beta_1$ where $\beta_1 = \beta - \alpha_2 M_2 g - \alpha_4 M_4 g - \alpha_6 M_6 g$. Substituting the values for α_2, α_4, α_6 we can solve for α similar to the case when two contiguous bits are revealed. Similarly we can solve when n contiguous bits of the key are known.

B. Algorithm

The generator g and the key are related as $\alpha\ g = \beta$
We know *n* contiguous bits of the key are known.
The key(=α) can be splited as follows

$$\alpha = \alpha_1 M_1 + \alpha_2 M_2 + \cdots + \alpha_{2n+1} M_{2n+1}.$$

Let $\alpha_2, \alpha_4, \ldots, \alpha_{2n}$ are known.

Algorithm to find α (key) from partial knowledge
Input: n know contiguous bits of the key i.e $\alpha[2]$, $\alpha[4]$, $\ldots, \alpha[2n]$ along with the position where they are occuring in the key.
Output: α , the key.
BEGIN
α can be written as
$\alpha = \alpha[1]M[1] + \alpha[2]M[2] + \cdots + \alpha[2n+1]M[2n+1]$
{ The values of M[ ] can be calculated as we know the positions of $\alpha[]$ }
$\beta_1 = \beta$
FOR $i = 1$ to n do
  $\beta_1 = \beta_1 - \alpha[2\times i]\times M[2\times i]\times g$
  $P_1 = P_2 = O$
END FOR
Arrays r and s are initialized to zeros
WHILE true **do**
  $P_1 = next(r, M, g, \beta_1, r_{old}, P_1);$
  $P_2 = next(s, M, g, \beta_1, s_{old}, P_2);$
  IF $P_1 == -P_2$ **then**
    FOR i in {1,3,…2n+1}
      $s_{old}[i] = -s_{old}[i]$
    END FOR
END IF
 IF $P_1 == P_2$ or $P_1 == -P_2$ **then**
  $r_{old}[1]M[1]g + r_{old}[3]M[3]g + \cdots + r_{old}[2n+1]M[2n+1]g\ \ \beta_1 r_{old}[2n+2]M[2n+2]g$
  $= s_{old}[1]M[1]g + s_{old}[3]M[3]g + \cdots + s_{old}[2n+1]M[2n+1]g + \beta_1 s_{old}[2n+2]M[2n+2]g$
  {We got an equation involving g and β (as β_1 can be expressed in β) so we solve to get α }
  Return α
 End if
End while
END
Procedure Next
Input: Integer array t,
  Integer array M,
  Generator g,
  β_1,
  Integer array t_{old},
  point on the curve
Output: Point
BEGIN
copy array t to t_{old}
$frac = p/($total unknown bits)
$temp = O$
$count =$ Number of unknown contiguous blocks.
for $i = 0$ to $count$ **do**
$temp = temp + t[i]\times M[i]\times g$
end for
$temp = temp - t[count+1]\times\beta_1$
x =x-coordinate of $temp$
$un = 0$
$flag = 0$
for $i = 0$ to $count$ **do**
  $un = un + i^{th}$ unknown bit length
  if $x \le un\times frac$ **then**
    $t[i] = t[i] + 1$
    $flag = 1$
    break the for loop;
  end if
end for
if $flag == 0$ **then**
$t[count+1] = t[count+1] + 1$
end if
$point = temp$
Return $point$
END PROCEDURE

VI. Comparison & Analysis

We have implemented Gopalakrishnan et al algorithm and our algorithm. C1, C2, .. are the names choosen for the curves. prime is the prime field

Two coefficients $a, b \in F_p$ that define the equation of the elliptic curve (i.e $y^2 = x^3 + ax + b$)
base point is the generator g of the curve
key is choosen (and it is alpha).
known bits are the bits given as the input of the programe along with the position information.

TABLE I. : INSTANCES OF DISCRETE LOG PROBLEMS

Curve	prime	a	b	base point	order of base point	key(in binary)	beta= key*base point
C1	104677	10	28	(59010, 27440)	52552	1010 0001 0111 01 (14 bits)	(78038, 56158)
C2	104683	23	84	(28451, 182)	104326	1010 0001 0111 01(14 bits)	(75645, 40063)
C3	1299919	37	837	(857959, 555867)	1299556	1100 0101 1110 1010 1 (17bits)	(972257, 858774)
C4	15485933	76	689	(15038473, 6300452)	7739768	1111 0100 1111 1000 0111 (20 bits)	(4036478, 979519)

Let the number of bits of the key be *l*. Let l_u be the number of unknown bits. Let l_k be the number of known bits. As $r_1(i)$, $r_3(i), \ldots,$ are incremented in the algorithm. When $b(r_1(i)) > b(\alpha_1)$ and $b(r_3(i)) > b(\alpha_3)$ and $b(r_5(i)) > b(\alpha_5)$, so on and $b(r')$ will be less than $\log p$, there is high possibility that we may get the match. So the time complexity is $O(\sqrt{2^{l_u}} \log p)$ in the average case. In the worst case it may go up to $O(\sqrt{p})$

Our method works when *n* number of contiguous bits of the key is revealed where as the earlier method [4] is applicable only when one contiguous bit is revealed.

TABLE II. PERFORMANCE OF GOPALAKRISHNAN ET AL ALGORITHM AND THE PROPOSED ALGORITHM

Name of the curve	known bits	position	no: of iterations goplakrishna et al algorithm	No: of iterations in our algorithm
C1	01 011	7	1555	660
C2	0111	9	3341	522
C3	10101	13	2411	1312
C4	11000	12	28147	5610

The time complexity of our method is $O(\sqrt{2^{(unknownbits)}})$ where as the time complexity of [5] is $O(\sqrt{2p} / N^{\frac{1}{4}})$ where $N = 2^{knownbits}$

VII. CONCLUSION AND FUTURE WORK

Gopalakrishnan et al. [4] presented an algorithm to solve the discrete logarithm problem in generic groups if a sequence of contiguous bits of the exponent is known. [4] converts the problem instance to another DLP problem and solves it using Pollard Rho method. The paper tries to generalize Goplakrishnan et al algorithm to solve the discrete logarithm problem. The proposed algorithm solves DLP when two or more contiguous bits of the key are revealed. The proposed algorithm tries to solve the DLP problem without converting it to an instance of another DLP problem.

Future Work

A better bound on the worst case time complexity can be derived.

ACKNOWLEDGEMENTS

N.Anil Kumar is supported by DST-CMS project Lr.No.SR/S4/MS:516/07, Dt.21-04-2008 and the support is gratefully acknowledged.

REFERENCES

[1] J. Cannon et al. The MAGMA computational algebra system. *Software available on line* `http://magma.maths.usyd.edu.au`, 2005.

[2] Darrel Hankerson, Affred Menezes, and Scott Vanstone. *Guide to Elliptic Curve Cryptography*. Springer Verlag, 2004.

[3] Andreas Enge. Elliptic Curves and Their Applications to Cryptography - An Introduction. Springer, 1999.

[4] K.Gopalakrishnan, Nicolas Theriault, and Chui Zhi Yao. Solving discrete logarithms from partial knowledge of the key. In K.Srinathan, C.Pandu Rangan, and M.Yung, editors, *Indocrypt 2007*, LNCS 4859, pages 224–237. Springer-Verlag Berlin, 2007.

[5] E. Prisner. *Graph Dynamics*. Pitman Research Notes In Mathematics Series. Pitman Research Notes In Mathematics Series, Longman House, Burnt Mill, Harlow, 1995.

[6] Quisquater,J.J., Samyde, D. Electromagnetic analysis (EMA): Measures and counter-measures for smart cards. In E-Smart 20101. LANCS, vol. 2140, pp 200-210. Springer Helidlberg (2010).

[7] L.C. Washington. Elliptic Curves: Number Theory and Cryptography. CRC Press, 2003.