IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 3, No. 1, May 2011
ISSN (Online): 1694-0814
www.IJCSI.org

A New Proxy Blind Signature Scheme based on ECDLP

Daniyal M. Alghazzawi[1], Trigui Mohamed Salim [2] and Syed Hamid Hasan[3]

**[1,2,3] Department of Information Systems,
King Abdul Aziz University, Kingdom of Saudi Arabia**

Abstract

A proxy blind signature scheme is a special form of blind signature which allows a designated person called proxy signer to sign on behalf of two or more original signers without knowing the content of the message or document. It combines the advantages of proxy signature, blind signature and multi-signature scheme and satisfies the security properties of both proxy and blind signature scheme. Most of the exiting proxy blind signature schemes were developed based on the mathematical hard problems integer factorization (IFP) and simple discrete logarithm (DLP) which take sub-exponential time to solve. This paper describes an secure simple proxy blind signature scheme based on Elliptic Curve Discrete Logarithm Problem (ECDLP) takes fully-exponential time. This can be implemented in low power and small processor mobile devices such as smart card, PDA etc. Here also we describes implementation issues of various scalar multiplication for ECDLP

Keywords: ECDLP, IFP, blind signature, proxy signature.

1. Introduction

Blind signature scheme was first introduced by Chaum [2]. It is a protocol for obtaining a signature from a signer, but the signer can neither learn the messages nor the signatures. The recipients obtain afterwards. In 1996, mamo et al proposed the concept of proxy signature [1]. In proxy signature scheme, the original signer delegates his signing capacity to a proxy signer who can sign a message submitted on behalf of the original signer. A verifier can validate its correctness and can distinguish between a normal signature and a proxy signature. A proxy blind signature scheme is a digital signature scheme that ensures the properties of proxy signature and blind signature. In a proxy blind signature, an original signer delegates his signing capacity to proxy signer.

2. Preliminaries

2.1 Notations

Common notations used in this paper as follows:

- p : The order of underlying finite field.
- F_p : the underlying finite field of order p
- *E:* elliptic curve defined on finite field F_p with large order.
- *G:* the group of elliptic curve points on *E*.
- *P:* a point in $E(F_p)$ with order *n*, where *n* is a large prime number.
- *H* (.): a secure one-way hash function.
- *d:* the secret key of the original signer *S* to be chosen randomly from [1, *n* - 1].
- *Q* is the public key of the original signer *S*, where *Q* = *d. Q*.
- *k:* Concatenation operation between two bit stings.

3. Backgrounds

In this section we brief overview of prime field, Elliptic Curve over that field and Elliptic Curve Discrete Logarithm Problem.

3.1 The finite field F_p

Let *p* be a prime number. The finite field F_p is comprised of the set of integers 0, 1, 2…. *p*-1 with the following arithmetic operations [4] [5] [6]:

- Addition: If $a, b \in F_p$, then $a + b = r$, where r is the remainder when $a + b$ is divided by p and $0 \leq r \leq p-1$. This is known as addition modulo p.
- Multiplication: If $a, b \in F_p$, then $a.b = s$, where s is the remainder when $a.b$ is divided by p and $0 \leq s \leq p-1$. This is known as multiplication modulo p.
- Inversion: If a is a non-zero element in F_p, the inverse of a modulo p, denoted a^{-1}, is the unique integer $c \in F_p$ for which $a.c = 1$.

3.2 Elliptic Curve over F_p

Let $p > 3$ be a prime number. Let $a, b \in F_p$ be such that $4a^3 + 27b^2 \neq 0$ in F_p. An elliptic curve E over F_p defined by the parameters a and b is the set of all solutions (x, y), $x, y \in F_p$, to the equation $y^2 = x^3 + ax + b$, together with an extra point O, the point at infinity. The set of points $E(F_p)$ forms an abelian group with the following addition rules [8]:

1. Identity : $P + O = O + P = P$, for all $P \in E(F_p)$
2. Negative: if $P(x; y) \in E(F_p)$ then $(x, y) + (x,-y) = O$, The point $(x,-y)$ is dented as $-P$ called negative of P.
3. Point addition: Let $P(x_1, y_1)$, $Q(x_2, y_2) \in E(F_p)$, then $P + Q = R \in E(F_p)$ and coordinate (x_3, y_3) of R is given by $x_3 = \lambda^2 - x_1 - x_2$ and $y_3 = \lambda(x_1 - x_3) - y_1$. Where $\lambda = {(y_2 - y_1)} \big/ {(x_2 - x_1)}$
4. Point doubling: Let $P(x_1, y_1) \in E(K)$ where $P \neq -P$ then $2P = (x3; y3)$ where $x_3 = ({3x_1^2 + a} \big/ {2y_1})^2 - 2x_1$ and $y_3 = {(3x_1^2 + a)} \big/ {2y_1} (x_1 - x_3) - y_1$.

3.3 Elliptic Curve Discrete Logarithm Problem (ECDLP)

Given an elliptic curve E defined over a finite field F_p ,a point $P \in E(F_p)$ of order n, and a point $Q \in \langle P \rangle$, find the integer $l \in [0, n-1]$ such that $Q = lP$. The integer l is called discrete logarithm of Q to base P, denoted $l = \log_p Q$ [8].

4. Proxy Signatures and Proxy Blind Signature

A proxy blind signature is a digital signature scheme that ensures the properties of proxy signature and blind signature schemes. Proxy blind signature scheme is an extension of proxy blind signature, which allows a single designated proxy signer to generate a blind signature on behalf of group of original signers. A proxy blind signature scheme consists of the following three phases[9]:

- Proxy key generation
- Proxy blind multi-signature scheme
- Signature verification

5. Security properties

The security properties for a secure blind multi-signature scheme are as follows [9]

- **Distinguishability**: The proxy blind multi-signature must be distinguishable from the ordinary signature.
- **Strong unforgeability**: Only the designated proxy signer can create the proxy blind signature for the original signer.
- **Non-repudiation**: The proxy signer can not claim that the proxy signer is disputed or illegally signed by the original signer.
- **Verifiability**: The proxy blind multi-signature can be verified by everyone. After verification, the verifier can be convinced of the original signer's agreement on the signed message.
- **Strong undeniably**: Due to fact that the delegation information is signed by the original signer and the proxy signature are generated by the proxy signer's secret key. Both the signer can not deny their behavior.
- **Unlinkability:** When the signer is revealed, the proxy signer can not identify the association between the message and the blind signature he generated.
- **Secret key dependencies**: Proxy key or delegation pair can be computed only by the original signer's secret key.
- **Prevention of misuse**: The proxy signer cannot use the proxy secret key for purposes other than

generating valid proxy signatures. In case of misuse, the responsibility of the proxy signer should be determined explicitly.

6. Proposed Protocol

The protocol involves three entities: Original signer *S*, Proxy signer P_s and verifier *V*. It is described as follows.

6.1 Proxy Phase

- **Proxy generation**: The original signer *S* selects random integer *k* in the interval [1, *n*-1]. Computes *R* = *k*. *P* = (x_1, y_1) and *r* = x_1 mod *n*. Where x_1 is regarded as an integer between 0 and *q* - 1. Then computes *s* = (*d* + *k*. *r*) mod *n* and computes $Q_p = s.P$.

- **Proxy delivery**: The original signer *S* sends (*s*, *r*) to the proxy signer P_s and make Q_p public.

- **Proxy Verification**: After receiving the secret key pairs (*s, r*), the proxy signer P_s checks the validity of the secret key pairs (*s, r*) with the following equation.

$$Q_p = s.P = Q + r.R \qquad (1)$$

6.2 Signing Phase

- The Proxy signer P_s chooses random integer *t* ∈ [1, *n*- 1] and computes *U* = *t* .*P* and sends it to the verifier *V*.
- After receiving the verifier chooses randomly α , β ∈ [1 *n*-1] and computes the following

$$\tilde{R} = U + \alpha.P - \beta.Q_p \qquad (2)$$

$$\tilde{e} = H(\tilde{R} \| M) \qquad (3)$$

$$e = (\tilde{e} + \beta) \bmod n \qquad (4)$$

and verifier *V* sends *e* to the proxy signer P_s.

- After receiving *e*, P_s computes the following

$$\tilde{s} = (t - s.e) \bmod n \qquad (5)$$

and sends it to *V*.

- Now *V* computes

$$s_p = (\tilde{s} + \alpha) \bmod n \qquad (6)$$

The tuples $(M, s_p, \tilde{e})$ is the proxy blind signature.

6.3 Verification Phase

The verifier *V* computes the following equation.

$$\gamma = H((s_p.P + \tilde{e}.Q_p) \| M) \qquad (7)$$

and verifies the validity of proxy blind signature $(M, s_p, \tilde{e})$ with the equality $\gamma = \tilde{e}$.

7 Security Analyses

7.1 Security Notions

Theorem 1 *It is infeasible for adversary A to derive signer's private key from all available public information.*

Proof: Assume that the adversary *A* wants to derive signer's private key *d* from his public key *Q*, he has to solve ECDLP problem which is computationally infeasible. Similarly, the adversary will encounter the same difficulty as she/he tries to obtain proxy signer's private key.

Theorem 2 *Proxy signature is distinguishable from original signer's normal signature.*

Proof: Since proxy key is different from original signer's private key and proxy keys created by different proxy signers are different from each other, any proxy signature is distinguishable from original signer's normal signature and different proxy signer's signature are distinguishable.

Theorem 3 *The scheme satisfies Unlinkability security requirement.*

Proof: In verification stage, the signer checks only whether $\gamma = H((s_p.P + \tilde{e}.Q_p) \| M)$ holds.

He does not know the original signer's private key and proxy signer's private key. Thus the signer knows neither the message nor the signature associated with the signature scheme.

8. Correctness

Theorem 4 *The proxy blind signature* $(M, s_p, \tilde{e})$ *is universally verifiable by using the system Public parameters.*

Proof: The proof of correctness of the signature is verified as follows. We have to prove that

$H((s_p.P+\tilde{e}.Q_p)\|M)=H(\tilde{R}\|M)$ *i.e.* to show

$$\begin{aligned}
&s_p.P+\tilde{e}.Q_p=\tilde{R}\\
&=(\tilde{s}+\alpha).P+\tilde{e}.Q_p\\
&=\tilde{s}.P+\alpha.P+\tilde{e}.Q_p\\
&=(t-s.e).P+\alpha.P+\tilde{e}.Q_p\\
&=t.P-(\tilde{e}+\beta).Q_p+\alpha.P+\tilde{e}.Q_p\\
&=t.P-\beta.Q_p+\alpha.P\\
&=U-\beta.Q_p+\alpha.P\\
&=\tilde{R}
\end{aligned}$$

9. Implémentation Issues

In this section we have discussed implementation issues, *i:e*. efficiency and size of the hard-ware. The basic operation for Cryptographic Protocols based on ECDLP; it is easily performed via repeated group operation. One can visualize these operations in a hierarchical structure. Point multiplication is at top level. At the next lower level is the point operations, which are closely related to coordinates used to represent the points. The lowest level consists of finite field operations such as addition, subtraction, multiplication and inversion.

9.1 Group Order

The order of the elliptic curve group over the underlying field is an important security parameter. There are attacks (for example Pohlig-Hellman attack) which can be launched on ECC if the group order is not divisible by a very large prime. In fact the Pohlig-Hellman attack dictates that the group order for ECC should be product of a large prime multiplied by a small positive integer less than 4. This small number is called *cofactor* of the curve. Various algorithms have been proposed in literature (for example Kedlaya's algorithm for ECC and Schoof's algorithm for ECC) for efficiently counting the group order. The group order of an elliptic curve is given by *Hasse's theorem*.

Theorem 5. *Let E be an elliptic curve over a finite field* F_p *of order q. Then the order* $\#E(F_p)$ *of the elliptic curve group is given by*

$$\#E(F_p)=q+1-t, where\,|t|\leq 2q^{1/2}$$

The parameter t is called trace of E over F_p . An interesting fact is that given any integer, there exists an elliptic curve E over F_p such that $\#E(F_p)=q+1-t$.

10. Point Representation and Cost of Group Operations

Point addition and point doubling are two important operations in ECC. Inversion in a finite field is an expensive operation. To avoid these inversions, several point representations have been proposed in literature. The cost of point addition and doubling varies depending upon the representation of the group elements. In the current section, we will briefly deal with some point representations commonly used. Let [*i*], [*m*], [*s*], [*a*] stand for cost of a field element inversion, a multiplication, a squaring and an addition respectively. Field element addition is considered to be a very cheap operation. In binary fields, squaring is also quite cheaper than a multiplication. If the underlying field is represented in normal basis then squaring is almost for free. Inversion is considered to be 8 to 10 times costlier than a multiplication in binary fields. In prime field the *I/M ratio* is even more. It is reported to be between 30 and 40 .

10.1 Elliptic Curves

Point representation in ECC is a well studied area. In the following two sections we describe some of the point representation popularly used in implementations. Table 1. Cost of Group Operations in ECC for Various Point Representations for Characteristic > 3

Coordinates	Cost (Addition)	Coordinates	Cost (Doubling)
$A+A\to A$	$1[i]+2[m]+1[s]$	$2A\to A$	$1[i]+2[m]+2[s]$
$P+P\to P$	$12[m]+2[s]$	$2P\to P$	$7[m]+3[s]$
$J+J\to J$	$12[m]+4[s]$	$2J\to J$	$6[m]+4[s]$
$C+C\to C$	$11[m]+3[s]$	$2C\to C$	$5[m]+4[s]$

Fields of Characteristic > 3 Elliptic curves over fields of characteristic > 3 have equations of the form $y^2=x^3+ax+b$. For such curves the following point representation methods are mostly used.

1. **In *Standard Projective Coordinates*** the curve has equation of the form
   $Y^2Z=X^3+aXZ^2+bZ^3$
   The point $(X:Y:Z)$, with $Z\neq 0$ in projective coordinates is the point (*X/Z*, *Y/Z*) in affine

coordinates. The point at infinity is represented by the point (0: 1: 0) and the inverse of (*X: Y: Z*) is the point (*X: -Y: Z*).

2. **In *Jacobian Projective Coordinates*** the curve has equation of the form
$Y^2Z = X^3 + aXZ^4 + bZ^6$.
The point, $Z \neq 0$ in Jacobian coordinates correspond to the affine point $(X/Z^2, Y/Z^3)$. The point at infinity is represented by the point (1: 1: 0) and the inverse of $(X:Y:Z)$ is the point $(X:-Y:Z)$. Point doubling becomes cheaper in Jacobian coordinates if the curve parameter *a* = -3.

3. **In *Chudonovski Jacobian Coordinates***, the Jacobian point $(X:Y:Z)$ is represented as $(X:Y:Z:Z^2:Z^3)$. Cost of point addition in Chudonovski Jacobian coordinates is the minimum among all representations.

In Table 1, we present the cost of addition and doubling in the coordinate systems described above. In the table we use A, P, J, C for affine, projective, Jacobian and Chudnovski Jacobian respectively. By $2A \to A$, we mean the doubling formula in which the input is in affine and so is the output. Similarly for addition and other coordinate systems.

Fields of Characteristic 2 We will consider only non-super singular curves. Elliptic curves (non-super singular) over binary fields have equations of the form $y^2 + xy = x^3 + ax^2 + b$. For such curves the following point representation methods are mostly used.

1. **In *Standard Projective Coordinates*** the curve has equation of the form
$Y^2Z + XYZ = X^3 + aX^2Z + bZ^3$
The point $(X:Y:Z)$, with $Z \neq 0$ in projective coordinates is the point (*X*=*Z*, *Y*=*Z*) in affine coordinates. The point at infinity is represented by the point (0: 1: 0) and the inverse of $(X:Y:Z)$ is the point (*X: X + Y: Z*).

2. **In *Jacobian Projective Coordinates*** the curve has equation of the form
$Y^2 + XYZ = X^3 + aX^2Z^2 + bZ^6$
The point $(X:Y:Z)$, with $Z \neq 0$ in Jacobian coordinates correspond to the affine point $(X/Z^2, Y/Z^3)$. The point at infinity is represented by the point (1 : 1 : 0) and the inverse of $(X:Y:Z)$ is the point (*X* : *X* + *Y* : *Z*).

3. **In *Lopez-Dahab Coordinates***, the point $(X:Y:Z)$, with $Z \neq 0$ represents the affine point $(X/Z, Y/Z^2)$. The equation of the elliptic curve in this representation is $Y^2 + XYZ = X^3Z + aX^2Z^2 + bZ^4$. The point at infinity is represented by the point (1 : 0 : 0) and the inverse of (*X* : *Y* : *Z*) is the point (*X* : *X* + *Y* : *Z*).

In Table 2 we present the cost of addition and doubling in the coordinate systems over binary fields. In the table we use *A*, *P*, *J* , *L* for affine, projective, Jacobian and Lopez-Dahab respectively. The table follows the same notational convention as in last subsection. Note that in Table 2 we have neglected squaring also. That is because in binary fields squaring is a much cheaper operation than multiplication, if one point is in affine and the other is in projective or some other weighted co-ordinate, then point addition becomes relatively cheaper. This operation is called *addition in mixed coordinates or mixed addition*. In ECC, the base point is generally stored in affine coordinates to take advantage of mixed additions.

Table 2. Cost of Group Operations in ECC for Various Point Representations in Even Characteristics

Coordinates	Cost (Addition)	Coordinates	Cost (Doubling)
$A+A \to A$	$1[i]+2[m]$	$2A \to A$	$1[i]+2[m]$
$P+P \to P$	$13[m]$	$2P \to P$	$7[m]+3[s]$
$J+J \to J$	$14[m]$	$2J \to J$	$5[m]$
$L+L \to L$	$14[m]$	$2L \to L$	$4[m]$

11. Scalar Multiplications

In ECC, computationally the most expensive operation is scalar multiplication. It is also very important from security point of view. The implementation attacks generally target the computation of this operation to break the cryptosystem. Given a point *X* and a positive integer *m*, computation of *m* × *X* = *X* + …… (*m* times)….. + *X* is called the operation of scalar multiplication. In this section we briefly outline various scalar multiplication algorithms proposed in literature. We do not include multi scalar multiplication methods (i.e. methods to compute ($lP + mQ$). Also, due to the vastness of the subject and space constraints we will elaborate only those methods which are discussed in depth in this dissertation. The basic algorithms to compute the scalar multiplication are the age old binary algorithms. They are believed to have been known to the Egyptians two thousand years ago. The two versions of DBL-AND-ADD algorithm are defined above. These algorithms invoke two functions ADD and DBL. ADD takes as input two points X_1 and X_2 and return their sum $X_1 + X_2$, DBL takes as input one point X and computes its double $2X$.

Algorithm DBL-AND-ADD (Left-to-right binary method)

Input: X, m $(m_{k-1}....m_1, m_0)$
Output: mX.

1. $E = m_{k-1}X$
2. *for i = k-2 down 0*
3. *E = DBL(E)*
4. *if* $m_i = 1$
5. *E = ADD(E, X)*
6. *return E*

Algorithm DBL-AND-ADD (Right-to-left binary method)

Input : X, m , $(m_{k-1}....m_1, m_0)$
Output : mX.

1. $E_0 = X, E_1 = 0$
2. *for i = 0 to k-1*
3. *if* $m_i = 1$
4. $E_1 = ADD(E_0, E_1)$
5. $E_0 = DBL(E_0)$
6. *return* (E_1)

Both the algorithms first convert the scalar multiplier m into binary. Suppose m has a n-bit representation with hamming weight h. Then, mX can be computed by n-1 invocations of DBL and h - 1 invocations of ADD. Hence cost of the scalar multiplication is $(n-1) \times \cos t(DBL) + h \times \cos t(ADD)$ .As the average value of h is n=2, on the average these algorithms require (n - 1) doubling and n=2 additions. As doublings are required more often than additions, attempts are made to reduce complexity of the doubling operation.

The scalar multiplication is the dominant operation in ECC. Extensive research has been carried out to compute it efficiently and a lot of results have been reported in literature. To compute the scalar multiplication efficiently there are three main approaches. As is seen in the basic binary algorithms the efficiency is intimately connected to the efficiency of ADD and DBL algorithms. So the first approach is to compute group operations efficiently. The second approach is to use a representation of the scalar such that the number of invocation of group operation is reduced. The third approach is to use more hardware support (like memory for pre-computation) to compute it efficiently. In some proposals these have approaches have been successfully combined to yield very efficient algorithms. As noted in the above, the cost of ADD and DBL depend to a large extent on the choice of underlying field and the point representation. Hence the cost of scalar multiplication also depends upon these choices. Based on the underlying field more efficient operations have been proposed. Over binary fields for ECC, using a point halving algorithm instead of DBL has been proved to be very efficient. Over fields of characteristic 3, point tripling has been more efficient. There are proposals for using fancier algorithms like the ones efficiently computing $2P + Q$, $3P + Q$ etc. instead of ADD and DBL.

12. Conclusions

The security of the scheme is hardness of solving ECDLP. The primary reason for the attractiveness of ECC over systems such as RSA and DSA is that the best algorithm known for solving the underlying mathematical problem namely, the ECDLP takes fully exponential time. In contrast, sub-exponential time algorithms are known for underlying mathematical problems on which RSA and DSA are based, namely the integer factorization (IFP) and the discrete logarithm (DLP) problems. This means that the algorithms for solving the ECDLP become infeasible much more rapidly as the problem size increases more than those algorithms for the IFP and DLP. For this reason, ECC offers security equivalent to RSA and DSA while using far smaller key sizes. The benefits of this higher-strength per-bit include higher speeds, lower power consumption, bandwidth savings, storage efficiencies, and smaller certificates. This can be implemented in low power and small processor mobile devices such as smart card, PDA etc. In this proposed scheme it is infeasible for adversary to derive signer's private key from all available public information. This protocol also achieves the security like requirements distinguishability, strong unforgeability, non-repudiation, and unlinkability.

References

[1]. M.Mambo, K.Usda and E.Okamoto Proxy signature: Delegation of power to sign messages "IEICE Transaction on Fundamentals", E79-A(1996), pp.1338-1353, 1996.

[2]. D.Chaum Blind Signature for Untraceable Payments, In Crypto 82, New York, Plenum Press, pp.199-203, 1983

[3]. S.J.Hwang and C.H.Shi A Simple multi-signature scheme, "Proceeding of 10th National conference on Information Security, Taiwan", 2000.

[4]. N. Koblitz. A course in Number Theory and Cryptography, 2nd edition Springer-Verlag-1994

[5]. K. H Rosen "Elementary Number Theory in Science and Communication", 2nd ed., Springer-Verlag, Berlin, 1986.

[6]. A. Menezes, P. C Van Oorschot and S. A Vanstone Handbook of applied cryptography. CRC, Press, 1997.

[7]. D. Hankerson, A .Menezes and S.Vanstone. Guide to Elliptic Curve Cryptography, Springer Verlag, 2004.

[8]. "Certicom ECC Challenge and The Elliptic Curve Cryptosystem"available http://www.certicom.com/index.php.

[9]. J.P.Kar Proxy Blind Multi-signature Scheme using ECC for handheld devices. Available at "International Association for Cryptology Research" http://eprint.iacr.org/2011/043.pdf .

Daniyal M. Alghazzawi has completed his Ph.D in Computer Science from University of Kansas in 2007, Master of Science in Teaching & Leadership in 2004 and Master of Science in Computer Science in 2003 from University of Kansas. He has worked as Web Programmer at ALTec (Advanced Learning Technologies) . Dr. Daniyal is currently Chairman of the Information Systems Department, Faculty of Computing and Information Technology, King Abdulaziz University. He has 04 journal papers and conferences to his credit. His research interest includes e-Security and Cryptography. Dr. Daniyal is a member of IEEE (Education Transaction) and ACM-SIGCSE (Special Interest Group in Computer Science Education) .

Trigui Mohamed Salim is currently working as a Lecturer at Information System Department, faculty of Computing and Information Technology, King Abdul Aziz University , KSA.. He has completed Master of Science (Information Technology) in 2009 from University, Utara Malaysia and Bachelor of Computer Science and Multimedia from University of Sfax, Tunisia in 2007. He has one conference paper to his credit. His research interest is e-Security and Cryptography.

Syed Hamid Hasan has completed his PhD in Computer Science from JMI, India, MSc in Statistics from AMU, India. Also he has completed Post-Graduate Diploma in Computer Science from the same university. Prof. Hamid has worked as a Head of Computer Science department at the AMU, India and was also Head of IT department at the Musana College of Technology, Sultanate of Oman. Dr Hamid is currently working as a Professor at Information Systems department, faculty of Computing and Information Technology, King Abdul Aziz University, Kingdom of Saudi Arabia. He was Reviewer for NDT 2009, Ostrava, Czech Republic, 2009. Co Sponsored by IEEE Communications Society. He is included in the Panel of referees of "The Indian journal of community health", was Chief Coordinator of the National Conference on "Vocationalization" of Computer Education" held on 28-29 September-1996 at A.M.U. Aligarh-India. He is a life Member of Indian Society for Industrial and Applicable Mathematics (ISIAM), Computer Society of India, Fellow National Association of Computer Educators & Trainers (FNACET), India. He has 20 research articles in conferences & journals to his credit. His research interest is e-Security and Cryptography.