ISSN:0975-9646

A Remote Smart Card Authentication Protocol Using Elliptic Curves

Ranbir Soram[1], Rajeev Chatterjee[2], Durga Prasad Roy[2], Rupesh Patidar[2]

[1]*Department of Computer Science & Engineering*
Manipur Institute of Technology, Takyelpat, Imphal-795001, India

[2]*Department of Computer Science & Engineering*
National Institute of Technical Teachers' Training & Research, Salt lake, Kolkata-700106, India

***Abstract*— A remote user authentication scheme is a client-server based protocol whereby a server identifies the identity of a remote client when it logs on to the server through unsecured network. This paper proposes a protocol to authenticate remote smart cards using elliptic curves. The proposed protocol has three phases- registration phase, login phase, and authentication phase. When a genuine user wants to login the computer system, he has to insert his smart card into the login device and keys in his identity, password and private keys.**

***Keywords*— Elliptic Curve, Smart Card, Cryptography, ECDLP**

I. INTRODUCTION

We live in an information age where information is treated as an asset that has a value like any other asset that we possess. So, we need to keep information secured from attacks and hackers. To keep information safe and secured it needs to be hidden from unauthorized access, protected from unauthorized modification and so on. Just a few decades from today, computer networks had been created and it has been creating a change in the use of information in the sense that information is distributed. It is now required to an authorized person to send and procure information from a far off place using computer networks. A new requirement has come up in the picture when the information is transmitted from one computer to another i.e., there should be a way to maintain its confidentiality on the way when it is transported from one computer to another in the network. So, the need for the public-key cryptography comes into picture.

In public-key cryptography, there are two keys:- a private key and a public key [16]. The private key is kept by the receiver. The public key is announced to the public. There are numerous public-key cryptography algorithms in the literature but many of these are found to be insecure and many are impractical to implement and use. As of now, only a few of those algorithms are considered both secure and practical. Of these secure and practical public-key algorithms, a few are suitable for encryption and still others are only useful for authentication. For example, RSA is presently used for both encryption and authentication [15]. It is very slow in actual practise. Elliptic Curve Cryptography is one of a few public-key algorithms that can be used in place of RSA. We begin with a discussion on Smart cards.

I. What is a smart card?

A smart card looks like a debit card in size and shape, but inside it is completely different as it contains a computer with a CPU and a memory [7]. The chip of a smart card contains a microprocessor, ROM, programmable ROM, and a small amount of Random Access Memory. A programmable ROM needs a larger volume than a PROM of the same size making programmable ROM size becomes an important factor for the price of a smart card. At present, most smart cards have an 8-bit microprocessor, but there are some smart cards which are incorporated with 16-bit or 32-bit processors running at 25 to 32 MHz [7]. An optional cryptographic coprocessor will enhance the performance of cryptographic operations. The beauty of having a microprocessor in cards is that by performing signature and decryption operations on the card itself, the user's private key never needs to leave the card. At the same time, the integration of smart cards into your system introduces its own security issues, as many people access card data in a variety of applications. The information stored in the ROM is written during production. It contains the card operating system and might also contain some applications. The programmable ROM is used for permanent storage of data but can be erased and rewritten again. Even if the smart card is unpowered, the programmable ROM still keeps the data.

II. AUTHENTICATION

In authentication, the identity of the entity or user is verified prior to access to the system resources or starting a transaction of data or value [16]. For example, a student who needs to access his university resources needs to be authenticated during the logging process. This is to protect the interests of the university and the student.

Remote user authentication using smart cards is a good solution for many applications. Smart card implementation ensures secure communications. Several schemes using timestamp for remote authentication have already been in use and discussed in the literatures [10]. A remote password authentication scheme authenticates the legitimacy of the remote user over insecure channel.

The ways in which someone may be authenticated fall into three categories, based on what are known as the factors of authentication: something you know, something you have, or something you are [16]. Each **authentication factor** covers a range of elements used to authenticate or verify a person's identity prior to being granted access, approving a transaction request, signing a document or other work product, granting authority to others, and establishing a chain of identity.

I. Something known: This is a secret known only to the claimant that can be checked by the verifier. Examples are a password, a PIN, and a secret key etc.
II. Something possessed: This is something that can prove the claimant's identity. Examples are a passport, an identity card, and a smart card.
III. Something inherent: This is the inherent characteristics of the claimant. Examples are conventional signatures, fingerprints, voices, and retina scan image.

III. SMART CARD SECURITY

As the issuer of card, you must define all of the desired parameters for card and system security. As mentioned above, a card or token-based system treats a card as an active computing device. The communication between the host and the card can be series of sub-communications to determine if the card is authorized to access the system. The process also checks if the user can be identified, authenticated in case the card present the appropriate credentials to initiate a transaction with the system. The card itself can also demand the same security requirements from the host before starting a transaction.

IV. THREATS TO CARDS AND DATA SECURITY

Taking into account the security system planning, there is a need for authorized users to access data reasonably easily and at the same time there are many threats that this access presents to the integrity and safety of the information. When analyzing the threats to your data, an organization should look closely at two specific areas: Internal attacks and External attacks [16]. The first common compromise of data comes from disgruntled employees. Knowing this, a good system manager separates all back-up data and back-up systems into a separately partitioned and secured space. The introduction of viruses and the attempt of formatting of network drives is typical internal attack behaviour.

External attacks are typically aimed at the weakest link in a company's security armour. A typical example of external attack is the interception of the transmission of your data in the Internet.

V. PROBLEM OF RSA IN SMART CARDS

The microprocessor on the smart card is there for security reason. The microprocessor enforces security mechanism while accessing the data on the card. If the host computer read and write the smart card's memory, it would be no different than a pen-drive. Most smarts cards may have up to 16 kilobytes of RAM, 128 kilobytes of ROM, 256 kilobytes of programmable ROM. Note that in smart card terminology, 1K means one thousand bits, not one thousand 8-bit characters. One thousand bits will normally store 128 characters - one sentence of text. However, with the development in modern data compression techniques, the amount of data stored on the smart card can be significantly increased beyond this limit.

One of the main problems of RSA is its demand for a huge key length to meet the challenges in today's security scenario. When you create an RSA key pair, you specify a key length in bits, as generally you would for other algorithms. Specifically, the key length of an RSA key specifies the **number of bits in the modulus**. But the million dollar question is "what RSA key length should we choose".

Experts say that an RSA key length of 1024 bits is sufficient for many medium-security purposes such as web site logins but for high-security applications such as online financial funds transfers or for data that needs to remain confidential for more than a few years; you should use at least a 2048-bit key and it can be confirmed using table 1. To keep data confidential for more than the next two decades, RSA experts recommend a key size larger than 2048 bits [16].

TABLE 1
RSA KEY LENGTHS OF SOME ORGANIZATIONS

Organization	RSA Key length
Google	1024
Yahoo	1024
SBI online	2048
eBay	2048
Union Bank of India	2048

A larger key increases the maximum number of bytes that we can encrypt at once, and also the security of the encryption. But it has a serious problem in practice. With every doubling of the RSA key length, decryption is about 8 times slower. The size of ciphertext also become huge considerably. The key length also affects the speed of *encryption*, which is slower by a factor of 4. In this situation, it is not at all possible to use RSA in smart cards. The use of Elliptic Curves may be a right choice in smart card. The comparisons in Table 2 demonstrate that smaller parameters can be used in elliptic curve cryptography (ECC) than with RSA system at a given security level. By a *security level* of k bits we mean that the best algorithm known for breaking the system takes approximately 2^k steps. The difference in parameter sizes is especially pronounced for higher security levels. The advantages that can be gained from smaller parameters include speed (faster computations) and smaller keys and certificates. At the 132-bit ECC/1024-bit RSA security level, an elliptic curve exponentiation for general curves over arbitrary prime fields is roughly 5 to 15 times as fast as an RSA private key operation, depending on the platform and optimizations. At the 256-bit ECC/3072-bit RSA security level the ratio has already increased to between 20 and 60, depending on optimizations. To secure a 256-bit AES key, ECC-521 can be expected to be on average 400 times faster than 15,360-bit RSA [22].

TABLE 2
RSA AND ECC KEY SIZES

Security level	**64**	**80**	**112**	**120**	**128**	**256**
ECC	106	132	185	237	256	512
RSA	512	1024	2048	2560	3072	15360

VI. ELLIPTIC CURVE

Elliptic curves are a specific class of algebraic curves. The "Weierstrass form" of an elliptic curve equation [1, 2]:-

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

The constant a_1, a_2, a_3, a_4, a_6 and the variables x, y can be complex, real, integers, polynomials, or even any other field elements. So, the mathematics of elliptic curve cryptography is so deep and complicated. But in practice we must specify which field, F, these constants and the variables, x, y belong to and $\Delta \neq 0$, where Δ is the discriminant of E and is defined as follows [1, 2]:-

$$\Delta = -d_2^2 d_8 - 8d_4^3 - 27d_6^2 + 9d_2 d_4 d_6$$
$$d_2 = a_1^2 + 4a_2$$
$$d_4 = 2a_4 + a_1 a_3$$
$$d_6 = a_3^2 + 4a_6$$
$$d_8 = a_1^2 a_6 + 4a_2 a_6 - a_1 a_3 a_4 + a_2 a_3^2 - a_4^2$$

We say that E is defined over K when the coefficients a_1, a_2, a_3, a_4, a_6 (and of course, the variables x and y) of the equations come from the elements of the field K. So, we sometimes write $E(K)$ to emphasize that E is defined over K, and K is called the underlying field. If E is defined over K, then E is also defined over any extension field of K.

A. ELLIPTIC CURVE OVER GALOIS FIELDS

Using the real numbers for cryptography has a lot of problem as it is very difficult to store them precisely in computer memory and predict how much storage will be needed for them. The difficulty can be solved by using Galois fields. In a Galois field, the number of elements is finite [16]. Since the number of elements if finite, we can find a unique representation for each of them, which allows us to store and handle the elements in an efficient way. Galois showed that the number of elements in a Galois field is always a positive prime power, and is denoted by $GF(p^n)$. Two special Galois fields are common in Elliptic Curve Cryptography. They are $GF(p)$ when $n = 1$ and $GF(2^n)$ when $p = 2$.

B. ELLIPTIC CURVE OVER PRIME GALOIS FIELDS

An elliptic group over a prime Galois Field uses a special elliptic curve of the form

$$y^2 (\text{mod } p) = x^3 + ax + b (\text{mod } p)$$

where $a, b \in GF(p), 0 \leq x \leq p$ and $-16(4a^3 + 27b^2) \bmod p \neq 0$. The constants ***a*** and ***b*** are non-negative integers smaller than the prime p. The condition that $-16(4a^3 + 27b^2) \bmod p \neq 0$ implies that the curve has no "singular points" [1 ,2].

C. GROUP LAW

The mathematical property that makes elliptic curves useful for cryptography is simply that if we take two (distinct) points on the curve, then the chord joining them intercepts the curve in a third point (because we have a cubic curve). If we then reflect that point in the x-axis we get another point on the curve (since the curve is symmetric about the x-axis). This allows us to define a form of arithmetic on the curve. Let E be an elliptic curve defined over the field **K**. There is a *chord-and-tangent rule* for adding two points in $E(K)$ to give a third point in $E(K)$. Take any two points on the curve; draw a line between them; and the negative of the third point, which intersects both the curve and the line, is the "sum" of the first two points. Together with this addition operation, the set of points $E(K)$ forms an abelian group with **0** serving as its identity [1, 2]. It is this group that is used in the construction of elliptic curve cryptographic systems. Algebraic formulae for the group law can be derived from the geometric description.

Group law for $y^2 = x^3 + ax + b$ over $GF(p)$.

I. Identity: $P + 0 = 0 + P = P$ for all $P \in E(K)$.

II. Negative: If $P = (x, y) \in E(K)$, then $(x, y) + (x, -y) = 0$. The point $(x, -y)$ is denoted by -P and is called the negative of P; note that -P is indeed a point in $E(K)$. Also, $-0 = 0$.

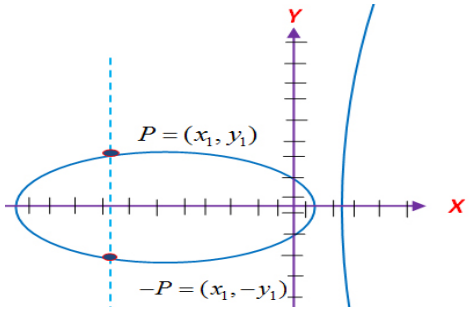
III. Point addition: Let $P = (x_1, y_1) \in E(K)$ and $Q = (x_2, y_2) \in E(K)$ where $P \neq \pm Q$ .Then $P + Q = R(x_3, y_3)$, where $x_3 = \lambda^2 - x_1 - x_2$, $y_3 = \lambda(x_1 - x_3) - y_1$ and $\lambda = \dfrac{y_2 - y_1}{x_2 - x_1}$

IV. Point doubling: Let $P = (x_1, y_1) \in E(K)$, where $P \neq \pm P$. Then $2P = R(x_3, y_3)$, where $x_3 = \lambda^2 - 2x_1$, $y_3 = \lambda(x_1 - x_3) - y_1$ and $\lambda = \dfrac{3x_1^2 + a}{2y_1}$

D. GEOMETRICAL INTERPRETATION OF GROUP LAW

I. Negative of a Point

Let's take a point $P = (x, y)$. The formula for finding $-P$ is $-P = (x, -y)$ as shown in the figure 1.



FGURE 1: NEGATIVE OF A POINT

II. Addition of two points

We can define the addition of any two points on an elliptic curve by drawing a line between the two points and finding the point at which the line intersects the curve. For the math to work, the negative of the intersection point is defined as the "elliptic sum" by mathematicians as shown in figure 2.

Mathematically we write:

$$R = P + Q.$$

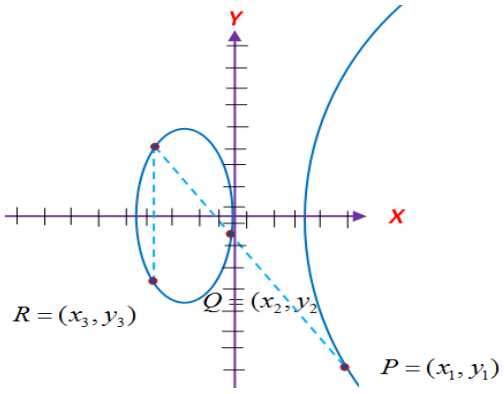


FIGURE 2 : ADDITION OF TWO POINTS

It turns out that this "addition" satisfies all the usual algebraic properties that we associate with integers, provided we define a single additional point "the point at infinity", which plays the role of 0 in the integers. In other words, we can define a form of arithmetic on the points of an elliptic curve (plus the point at infinity) that lends itself to normal algebraic manipulation. In mathematical terms, we can define a finite additive abelian group on the points of the curve, with the zero being the point at infinity [2].

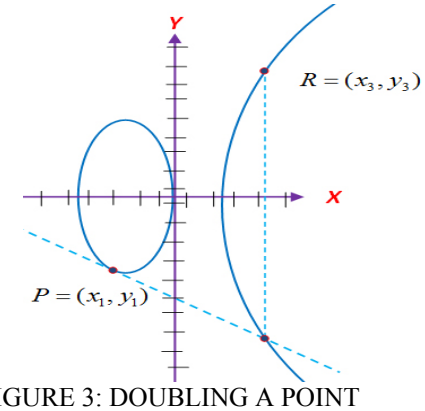


FIGURE 3: DOUBLING A POINT

III. Doubling of a point

If $P=(x_1, y_1)$, then the *double* of *P*, denoted by, $R=(x_3, y_3)$, is defined as follows. First draw the tangent line to the elliptic curve at *P*. This line intersects the elliptic curve in a second point. Then *R* is the reflection of this point in the x -axis. This is depicted in figure 3. We can extend this idea to define $P+P+P=3P$, and extending this idea further, we can define $P+P+P+\ldots+k$ times $=kP$, for any integer *k*, and hence define the order of P, being the smallest integer k such that $kP=0$, where 0 denotes the point at infinity[16]. Figure 4 shows some multiples of $P=(-1,-2)$ on the curve $y^2=x^3-5x$.

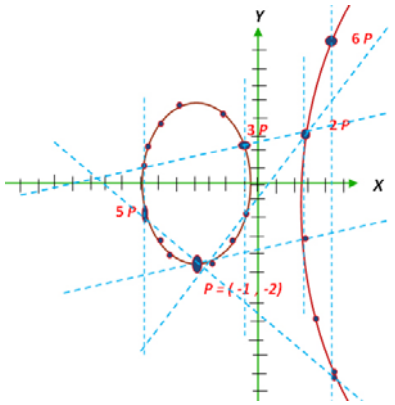


FIGURE 4: SOME MULTIPLES OF $P=(-1,-2)$.

To elucidate doubling of a point, consider the elliptic curve

$$y^2 = x^3 + x + 4$$

defined over $GF(23)$. This curve is represented by $E_{23}(1,4)$. We first note that $4a^3+27b^2 = 4+432 = 436 \equiv 22 (\text{mod } 23) \neq 0 (\text{mod } 23)$. The points in $E_{23}(1,4)$ are the following [2]:-

TABLE 3

POINTS ON THE CURVE $E_{23}(1,4)$

0	*(0,2)*	*(0,21)*	*(1,11)*	*(1,12)*	*(4,7)*
(4,16)	*(7,3)*	*(7,20)*	*(8,8)*	*(8,15)*	*(9,11)*
(9,12)	*(10,5)*	*(10,18)*	*(11,9)*	*(11,14)*	*(13,11)*
(13,12)	*(14,5)*	*(14,18)*	*(15,6)*	*(15,17)*	*(17,9)*
(17,14)	*(18,9)*	*(18,14)*	*(22,5)*	*(22,19)*	

Let $P=(4,7)$ and $Q=(13,11)$. Then $P+Q=R(x_3, y_3)$ is computed as follows-

$$\lambda = \frac{11-7}{13-4} = \frac{4}{9} = 4 \text{X} 9^{-1} (\text{ mod } 23) = 4 \text{X} 18 (\text{ mod } 23) = 72 \text{ mod } 23 = 3$$

$$x_3 = 3^2 - 4 - 13 = -8 \equiv 15 (\text{mod } 23), \text{ and}$$

$$y_3 = 3(4-15) - 7 = -40 \equiv 6 (\text{mod } 23)$$

Hence, $R=(15,6)$.

Again, let $P=(4,7)$. Then $2P=P+P$ is calculated as follows:-

$$\lambda=\left(\frac{3\text{X}4^2+1}{14}\right)=49\text{X}14^{-1}=49\text{X}5=245\ (\text{mod}\ 23)=15$$

$x_3=15^2-8=217\equiv 10(\text{mod}\,23)$ and

$y_3=15(4-10)=-97\equiv 18(\text{mod}\,23).$

Hence, $2P=(10,18).$

E. ELLIPTIC CURVE OVER $GF(2^n)$.

Let's look at elliptic curves over $GF(2^n)$.That means our constants are either polynomial or normal basis numbers. It also means we cannot use the simplified version of equation, which we used for integer numbers, for our elliptic curve equations.

The mathematicians tell us that we need to use either this version:

$$y^2+xy=x^3+ax^2+b \qquad (1)$$

or this version

$$y^2+y=x^3+ax+b \qquad (2)$$

But, the mathematicians say that the second form above, equation (2), is called a "supersingular" curve. These forms have the advantage that they can be computed quickly. However, being a special class of curves, they have some very special properties. These properties make supersingular curves unsuitable for cryptography.

The curves of equation (1) are called "nonsupersingular." To date, no method of attack is known to be less than fully exponential in time. Curves of this form are excellent for cryptographic applications. One must be careful in choosing the coefficients to get maximum benefit of security. A poor choice can create a curve that is easier for the cryptanalyst to attack. For equation (1) to be valid, *b* must never be 0. However, *a* can be 0. The rules are the same as before: Take any two points on the curve; draw a line between them; and the negative of the third point, which intersects both the curve and the line, is the "sum" of the first two points. Here we give the group laws of the first form of the curve[1,2].

Group law for $y^2+xy=x^3+ax^2+b$ over $GF(2^n)$

I. Identity: $P+0=0+P=P$ for all $P\in E.$

II. Negative: If $P=(x,y)\in E,$ then $(x,y)+(x,x+y)=0.$ The point $(x,x+y)$ is denoted by -P and is called the negative of P; note that -P is indeed a point in E. Also, $-0=0.$

III. Point addition: Let $P=(x_1,y_1)\in E$ and $Q=(x_2,y_2)\in E$ where $P\neq\pm Q$ .Then $P+Q=R(x_3,y_3)$, where

$$x_3=\lambda^2+\lambda+x_1+x_2+a \quad \text{and}$$

$$y_3=\lambda(x_1+x_3)+x_3+y_1 \text{ with}$$

$$\lambda=\frac{y_2+y_1}{x_2+x_1}$$

IV. Point doubling: Let $P=(x_1,y_1)\in E,$ where $P\neq -P$. Then $2P=R=(x_3,y_3),$ where $x_3=\lambda^2+\lambda+a$ and $y_3=x_1^2+\lambda x_3+x_3$ with

$$\lambda=x_1+\frac{y_1}{x_1}$$

Let us take an elliptic curve [16] $y^2+xy=x^3+g^3x^2+1$ over $GF(2^3)$ under the irreducible polynomial $f(x)=x^3+x+1.$ Here the generator, g, satisfies the relation $g^3+g+1=0$ or $g^3=g+1$ as the arithmetic is over $GF(2).$ The following table 4 shows the values of $g's$ and the points on the curve are given in table 5.

TABLE 4
POSSIBLE VALUES OF g's

0	*000*	$g^3=g+1$	*011*
1	*001*	$g^4=g^2+g$	*110*
g	*010*	$g^5=g^2+g+1$	*111*
g^2	*100*	$g^6=g^2+1$	*101*

TABLE 5
POINTS ON THE GIVEN CURVE

0	$(0,1)$	$(g^2,1)$	(g^2,g^6)
(g^3,g^2)	(g^3,g^5)	$(g^5,1)$	(g^5,g^4)
(g^6,g)	(g^6,g^5)		

Let $P=(0,1)$ and $Q=(g^2,1).$ We have $P+Q=R=(x_3,y_3)$ is computed as follows.

$$\lambda=\frac{1+1}{g^2+0}=0$$

$x_3=\lambda^2+\lambda+x_1+x_2+a=0+0+0+g^2+g^3=g^5.$

and

$y_3=\lambda(x_1+x_3)+x_3+y_1=0(0+g^5)+g^5+1$

$=g^5+1=g^2+g=g^4.$

So, $R=(g^5,g^4)=(111,110).$

Again take $P = (g^2, 1)$. $2P = P + P = R(x_3, y_3)$.

$$\lambda = g^2 + \frac{1}{g^2} = g^2 + g^5 = g + 1 = g^3$$

$$x_3 = \lambda^2 + \lambda + a = g^6 + g^3 + g^3 = g^6.$$

and

$$y_3 = x_1^2 + \lambda x_3 + x_3$$
$$= g^4 + g^9 + g^6 = g^4 + g^2 + (g^2 + 1)$$
$$= g^4 + 1 = (g^2 + g) + 1 = g^5.$$

Therefore, $R = (x_3, y_3) = (g^6, g^5) = (101, 111)$.

F. HASSE THEOREM AND POINT COUNTING

Let *E* be an elliptic curve defined over F_q. The number of points in $E(F_q)$, denoted $\#E(F_q)$, is called the *order* of *E* over F_q. Then Hasse's theorem says that the order of $E(F_q)$ satisfies the inequality [1,2]

$$q + 1 - 2\sqrt{q} \le \#E(F_q) \le q + 1 + 2\sqrt{q}.$$

An alternate formulation of Hasse's theorem is the following: if *E* is defined over F_q, then $\#E(F_q) = q + 1 - t$ where $|t| \le 2\sqrt{q}$; *t* is called the *trace* of *E* over F_q. Since $2\sqrt{q}$ is small relative to *q*, we have $\#E(F_q) \approx q$.

There are several methods presently known that can quickly determine the order of $E(F_q)$. Unfortunately none of them is effective once q is very large. An alternative approach is to use the order of certain points in $E(F_q)$. Since $E(F_q)$ is a group, and then the order of any point in $E(F_q)$ must divide $|E(F_q)|$, by Lagrange's theorem. In Hasse's theorem, we know that $|E(F_q)|$ is bounded in an interval of length $4\sqrt{q}$. If we can find a point in $E(F_q)$ of order $m > 4\sqrt{q}$, then there will be only one multiple of m lying in that interval, which must be $|E(F_q)|$. For example, let E be the elliptic curve $y^2 = x^3 - 10x + 21$ over $GF(557)$. It can be shown that the point (2, 3) has order 189. Hasse's theorem says that

$$557 + 1 - 2\sqrt{557} \le |E(F_{557})| \le 557 + 1 + 2\sqrt{557}$$

$$\text{i.e, } 511 \le |E(F_{557})| \le 605$$

But the only multiple of 189 in this interval is 3 as $3\text{X}189 = 576$. Hence, $|E(F_{557})| = 567$.

G. SUPERSINGULAR CURVES

Elliptic curves defined over a finite field are of two types. Most are what are called ordinary or non-supersingular curves, but a small number are supersingular[1]. As mentioned in section VI, the order or cardinality of an elliptic curve is $\#E(F_q) = q + 1 - t$, where $|t| \le 2\sqrt{q}$. Let *p* be the characteristic of F_q. An elliptic curve *E* defined over F_q is supersingular if *p* divides *t*, where *t* is the trace. If *p* does not divide *t*, then *E* is non-supersingular [2]. The problem with the supersingular elliptic curve is that the ECDLP in an elliptic curve E defined over a field F_q can be reduced to the ordinary DLP in the multiplicative group of some finite extension field of $F_q k$ for some $k \ge 1$. It follows that the reduction of ECDLP to ordinary DLP can be solved in a sub-exponential time, thus, compromising security of the system. To ensure that the reduction does not apply to a particular curve, one need to make sure that n, the order of the point P, does that divide $q^k - 1$ for small k.

H. AN IMPORTANT THEOREM

Let *E* be an elliptic curve defined over F_q. Then $E(F_q)$ is isomorphic to $Z_{n1} \oplus Z_{n2}$ where n_1 and n_2 are uniquely determined positive integers such that n_2 divides both n_1 and $q - 1$. Note that $\#E(F_q) = n_1 n_2$. If $n_2 = 1$, then $E(F_q)$ is a cyclic group. If $n_2 > 1$, then $E(F_q)$ is said to have rank *2*. If n_2 is a small integer (e.g., *n* = 2 ,3 or 4), we sometimes say that $E(F_q)$ is almost cyclic[1, 2, 12]. Since n_2 divides n_1 and $q - 1$, one expects that $E(F_q)$ is cyclic or almost cyclic for most elliptic curves *E* over F_q.

I. SECURITY OF ECC

Let *E* be an elliptic curve defined over a finite field and let, *P* be a point (called base point) on *E* of order *n* and k is a scalar. Calculating the point $Q = kP$ from P is very easy and $Q = kP$ can be computed by repeated point additions of P. However, it is very hard to determine the value of *k* knowing the two points: kP *and* P. This lead leads to the definition of Elliptic Curve Logarithm Problem (ECDLP) [12], which is defined as: "*Given a base point P and the point* $Q = kP$*, lying on the curve, find the value of scalar k, provided that such an integer exists*". The integer *k* is called the *elliptic curve discrete logarithm of Q to the base P*, denoted as $k = \log_P Q$.

VII. PROPOSED AUTHENTICATION PROTOCOL

Before we explain our protocol, we give a few important notations used in this section.

Alice:	The remote user.
Bob:	The authentication server.
$ID_{AB}(x, y)$:	Identity of Alice and is a point on Bob's curve.

$PW_{AB}(x,y)$: Password of Alice and is a point on Bob's curve.

$f : R_A(X,Y) \rightarrow R_B(X,Y)$: A mapping function that is used to map a point from Alice's curve to a point on Bob's curve.

$\oplus$: The concatenation operator.

The proposed protocol has three phases, registration phase, login phase, and authentication phase and each of them is explained below.

A. REGISTRATION PHASE

We use two different curves in this protocol. One curve is used by Alice and the other curve is used by Bob. Each one of them exchanges with each other the curve parameters $D = (q, FR, a, b, G, n, h)$ comprising of the following:-

q	is the order of the field used
FR	Field representation used for elements of F_q.
a, b	a, b in $y^2 = x^3 + ax + b$
G	base point of the curve
n	order of the base point
h	cofactor and $h = \frac{\#E(F_q)}{n}$

The exchange can take place in an unsecure medium as the curves are public.

Bob chooses using his curve a private key d_B such that $d_B \in [1, n_B - 1]$ and a public key $Q_B(x,y) = d_B.G_B(x,y)$.

Alice chooses m integers $d_{A1}, d_{A2}, ..., d_{Am} \in [1, n_A - 1]$ as his private keys. He then chooses m points $P_{A1}(x,y), P_{A2}(x,y), ...,$ and $P_{Am}(x,y)$ on his curve. Then, he calculates a point

$$P_D(x,y) = \sum_{i=1}^{m} d_{Ai}.P_{Ai}(x,y)$$

$$= d_{A1}.P_{A1}(x,y) + d_{A2}.P_{A2}(x,y) + ... + d_{Am}.P_{Am}(x,y).$$

His public key is the tuple $\left(P_D(x,y), P_{A1}(x,y), P_{A2}(x,y), ..., P_{Am}(x,y)\right)$.

Alice submits his public key $\left(P_D(x,y), P_{A1}(x,y), P_{A2}(x,y), ..., P_{Am}(x,y)\right)$ to Bob for registration. Bob calculates Alice's identity and password as

$$ID_{AB}(x,y) = f\left(P_D(x,y)\right)$$

$$PW_{AB}(x,y) = d_B.ID_{AB}(x,y).$$

Then Bob issues to Alice a smart card which contains the public parameter $ID_{AB}(x,y)$. This value is unique for every user, and maintained by Bob. Bob also despatches $PW_{AB}(x,y)$ to Alice through a secure channel.

B. LOGIN PHASE

When Alice wants to login to Bob, he inserts his smart card into a card reader and keys $ID_{AB}(x,y)$, $PW_{AB}(x,y)$ and his private keys. Then smart card reader and Alice will perform the following steps:

I. Generates m random numbers r_{A1}, $r_{A2}, ..., r_{Am}$ and calculates a point

$$P_R(x,y) = \sum_{i=1}^{m} r_{Ai}.P_{Ai}(x,y)$$

II. Send the login request message $\left(ID_{AB}(x,y), \quad P_R(x,y)\right)$ to Bob.

III. Then Bob calculates an integer

$$e_B = g\left(T_{CB} \oplus ID_{AB}(x,y) \oplus PW_{AB}(x,y)\right).$$

Here T_{CB} is the current timestamp of Bob. Bob sends e_B to Alice.

IV. Alice keys in his private keys, password and calculates the followings:-

$$I) \quad x_{A1} = r_{A1} + e_B.d_{A1}$$

$$II) \quad x_{A2} = r_{A2} + e_B.d_{A2}$$

$$III) \quad ...$$

$$IV) \quad x_{Am} = r_{Am} + e_B.d_{Am}$$

$$V) \quad C_A(x,y) = e_B.ID_{AB}(x,y)$$

$$VI) \quad D_A(x,y) = C_A(x,y) + e_B.PW_{AB}(x,y)$$

$$VII) \quad t_A = g\left(T_{CA} \oplus e_B \oplus PW_{AB}(x,y)\right)$$

$$VIII) \quad E_A(x,y) = t_A.G_A(x,y)$$

V. Alice sends the tuple $\begin{pmatrix} x_{A1}, x_{A2}, ..., x_{Am}, C_A(x,y), \\ ID_{AB}(x,y), D_A(x,y), E_A(x,y), T_{CA} \end{pmatrix}$ to Bob.

Here T_{CA} is the current timestamp of the Alice.

C. AUTHENTICATION PHASE

Bob receives the login request and performs the following steps:

I. Check whether $ID_{AB}(x,y)$ is a valid user identity, if not, then Bob rejects the login request.

II. Check, whether $\left(T'_{CB} - T_{CA}\right) \leq \Delta T$, where T'_{CB} is current timestamp and ΔT is the permissible transmission delay. If ΔT is not reasonable, then Bob rejects the login.

III. Bob calculates

$$t_B = g\left(T_{CA} \oplus e_B \oplus PW_{AB}(x,y)\right),$$

where T_{CA} is the timestamp sent by Alice.

IV. Evaluate the following equations

$$I.\quad P_R(x,y)=\left(\sum_{i=1}^{m} x_{Ai}.P_{Ai}(x,y)\right)-e_B.P_D(x,y) \quad (3)$$

$$II.\quad D_A(x,y)-d_B C_A(x,y)=C_A(x,y) \quad (4)$$

$$III.\quad E_A(x,y)-t_B.G_A(x,y)=0 \quad (5)$$

If any of the above equations is not satisfied, then login is rejected otherwise login is allowed.

V. If the login request is rejected three times then automatically the user account is locked for the day.

D. PROOF THAT THE ALGORITHM WORKS

Rewriting equation (3), we have,

$$P_R(x,y)=\left(\sum_{i=1}^{m} x_{Ai}.P_{Ai}(x,y)\right)-e_B.P_D(x,y)$$
$$=x_{A1}.P_{A1}(x,y)+x_{A2}.P_{A2}(x,y)+\ldots+x_{Am}.P_{Am}(x,y)-e_B.P_D(x,y)$$

$$RSH$$
$$=x_{A1}.P_{A1}(x,y)+x_{A2}.P_{A2}(x,y)+\ldots+x_{Am}.P_{Am}(x,y)-e_B.P_D(x,y)$$
$$=\left(r_{A1}+e_B.d_{A1}\right)P_{A1}(x,y)+\left(r_{A2}+e_B.d_{A2}\right)P_{A2}(x,y)+\ldots+\left(r_{Am}+e_B.d_{Am}\right)P_{Am}(x,y)-e_B\left(\sum_{i=1}^{m} d_{Ai}.P_{Ai}(x,y)\right)$$

$$\left[\because x_{A1}=r_{A1}+e_B.d_{A1},\ x_{A2}=r_{A2}+e_B.d_{A2}\right]$$
$$\left[\because \quad \ldots \quad\right]$$
$$\left[\because x_{Am}=r_{Am}+e_B.d_{Am}\right]$$
$$\left[\because P_D(x,y)=\sum_{i=1}^{m} d_{Ai}.P_{Ai}(x,y)\right]$$

$$=r_{A1}P_{A1}(x,y)+e_B.d_{A1}P_{A1}(x,y)+r_{A2}P_{A2}(x,y)+e_B.d_{A2}P_{A2}(x,y)+\ldots+r_{Am}P_{Am}(x,y)+e_B.d_{Am}P_{Am}(x,y)-(e_B d_{A1}.P_{A1}(x,y)+e_B d_{A2}.P_{A2}(x,y)+\ldots+e_B d_{Am}.P_{Am}(x,y))$$
$$=r_{A1}P_{A1}(x,y)+r_{A2}P_{A2}(x,y)+\ldots+r_{Am}P_{Am}(x,y)+e_B.d_{A1}P_{A1}(x,y)+e_B.d_{A2}P_{A2}(x,y)+\ldots+e_B.d_{Am}P_{Am}(x,y)-e_B d_{A1}.P_{A1}(x,y)-e_B d_{A2}.P_{A2}(x,y)-\ldots-e_B d_{Am}.P_{Am}(x,y)$$
$$=r_{A1}P_{A1}(x,y)+r_{A2}P_{A2}(x,y)+\ldots+r_{Am}P_{Am}(x,y)$$
$$=P_R(x,y)$$
$$=LSH$$

Rewriting equation (4), we have,

$$D_A(x,y)-d_B C_A(x,y)=C_A(x,y)$$
$$LSH$$
$$=D_A(x,y)-d_B C_A(x,y)$$
$$=C_A(x,y)+e_B.PW_{AB}(x,y)-d_B e_B.ID_{AB}(x,y)$$
$$\left[\because D_A(x,y)=C_A(x,y)+e_B.PW_{AB}(x,y)\right]$$
$$=C_A(x,y)+e_B d_B.ID_{AB}(x,y)-d_B e_B.ID_{AB}(x,y)$$
$$\left[\because PW_{AB}(x,y)=d_B.ID_{AB}(x,y)\right]$$
$$=C_A(x,y)$$
$$=RSH$$

Again rewriting equation (5), we get,

$$E_A(x,y)-t_B.G_A(x,y)=0$$
$$LSH$$
$$=E_A(x,y)-t_B.G_A(x,y)$$
$$=t_A.G_A(x,y)-t_B.G_A(x,y)$$
$$\left[\because E_A(x,y)=t_A.G_A(x,y)\right]$$
$$=g\left(T_{CA}\oplus e_B\oplus PW_{AB}(x,y)\right).G_A(x,y)-g\left(T_{CA}\oplus e_B\oplus PW_{AB}(x,y)\right).G_A(x,y)$$
$$\left[\because t_A=g\left(T_{CA}\oplus e_B\oplus PW_{AB}(x,y)\right)\right]$$
$$\left[\because t_B=g\left(T_{CA}\oplus e_B\oplus PW_{AB}(x,y)\right)\right]$$
$$=0$$
$$=RHS$$

E. CONDITIONS UNDER WHICH THE ALGORITHM WILL WORK

From section VII, we know that Alice's public key is

$$\left(P_D(x,y),\ P_{A1}(x,y),\ P_{A2}(x,y),\ldots,P_{Am}(x,y)\right)$$

where

$$P_D(x,y)=\sum_{i=1}^{m} d_{Ai}.P_{Ai}(x,y)$$
$$=d_{A1}.P_{A1}(x,y)+d_{A2}.P_{A2}(x,y)+\ldots+d_{Am}.P_{Am}(x,y)$$

Let

$$P_{A1}(x,y)=\lambda_1 G_A(x,y)$$
$$P_{A2}(x,y)=\lambda_2 G_A(x,y)$$
$$\ldots$$
$$P_{Am}(x,y)=\lambda_m G_A(x,y),\ \text{where } \lambda_i \text{ are some integers and } G_A(x,y) \text{ is the generator of the Alice's curve.}$$

$$\therefore P_D(x,y)=\sum_{i=1}^{m} d_{Ai}.P_{Ai}(x,y)$$
$$=d_{A1}.P_{A1}(x,y)+d_{A2}.P_{A2}(x,y)+\ldots+d_{Am}.P_{Am}(x,y)$$
$$=d_{A1}\lambda_1 G_A(x,y)+d_{A2}\lambda_2 G_A(x,y)+\ldots+d_{Am}\lambda_m G_A(x,y)$$
$$=\left(d_{A1}\lambda_1+d_{A2}\lambda_2+\ldots+d_{Am}\lambda_m\right)G_A(x,y)$$

As given in section VII, the order of the Elliptic Curve of Alice is n_A, then Alice will have to make sure that the following inequality is satisfied.

$$\left(d_{A1}\lambda_1 + d_{A2}\lambda_2 + ... + d_{Am}\lambda_m\right) \le (n_A - 1).$$

But, in most cases, it will not create any problem as the order of any curve used in cryptography is very very huge

.

F. SECURITY ANALYSIS

It is assumed here that Eve steals the smart card of a person. He inserts the smart card into the card reader and keys $ID_{AB}(x,y), PW_{AB}(x,y)$ and private keys. Then smart card reader and he will perform the following steps:

I. Generates m random numbers r_{A1}, $r_{A2}, ..., r_{Am}$ and calculates

$$P_R(x,y) = \sum_{i=1}^{m} r_{Ai}.P_{Ai}(x,y)$$
$$= r_{A1}.P_{A1}(x,y) + r_{A2}.P_{A2}(x,y) + ... + r_{Am}.P_{Am}(x,y)$$

II. Send the login request message $\left(ID_{AB}(x,y), \quad P_R(x,y)\right)$ to Bob.

III. Then Bob calculates an integer $e_B = g\left(T_{CB} \oplus ID_{AB}(x,y) \oplus PW_{AB}(x,y)\right)$ and send it to him.

IV. Eve keys in private keys, password and calculates the followings:-

$I)$ $x_{A1} = r_{A1} + e_B.d_{A1}$

$II)$ $x_{A2} = r_{A2} + e_B.d_{A2}$

$III)$ ...

$IV)$ $x_{Am} = r_{Am} + e_B.d_{Am}$

$VI)$ $C_A(x,y) = e_B.ID_{AB}(x,y)$

$VI)$ $D_A(x,y) = C_A(x,y) + e_B.PW_{AB}(x,y)$

$VII)$ $t_A = g\left(T_{CE} \oplus e_B \oplus PW_{AB}(x,y)\right)$

$VIII)$ $E_A(x,y) = t_A.G_A(x,y)$

V. The tuple $(x_{A1}, x_{A2}, ..., x_{Am}, C_A(x,y), ID_{AB}(x,y), D_A(x,y), E_A(x,y), T_{CE})$ is sent to Bob by Eve. Here T_{CE} is the current timestamp of the Eve.

G AUTHENTICATION PHASE

Bob receives the login request and performs the following steps:

I. Check whether $ID_{AB}(x,y)$ is a valid user identity, if not, then Bob rejects the login request.

II. Check, whether $\left(T_{CB} - T_{CE}\right) \le \Delta T$, where T_{CB} is current timestamp and ΔT is the permissible transmission delay. If ΔT is not reasonable, then Bob rejects the login.

III. Bob calculates $t_B = g\left(T_{CE} \oplus e_B \oplus PW_{AB}(x,y)\right)$, where T_{CE} is the timestamp sent by Eve.

IV. Now Eve gets the following equations

$$I) \quad P_R(x,y) = \left(\sum_{i=1}^{m} x_{Ai}.P_{Ai}(x,y)\right) - e_B.P_D(x,y)$$
$$= x_{A1}.P_{A1}(x,y) + x_{A2}.P_{A2}(x,y) + ... + x_{Am}.P_{Am}(x,y) - e_B.P_D(x,y)$$

$$II) \quad D_A(x,y) - d_B C_A(x,y) = C_A(x,y)$$

$$III) \quad E_A(x,y) - t_B.G_A(x,y) = 0$$

From the above equations, let see whether Eve can recover the private keys and password in polynomial time. Consider the equation

$$P_R(x,y) = \left(\sum_{i=1}^{m} x_{Ai}.P_{Ai}(x,y)\right) - e_B.P_D(x,y)$$
$$= x_{A1}.P_{A1}(x,y) + x_{A2}.P_{A2}(x,y) + ... + x_{Am}.P_{Am}(x,y) - e_B.P_D(x,y)$$

We have

$x_{A1} = r_{A1} + e_B.d_{A1}$

$x_{A2} = r_{A2} + e_B.d_{A2}$

...

$x_{Am} = r_{Am} + e_B.d_{Am}$

$$P_R(x,y) = \sum_{i=1}^{m} r_{Ai}.P_{Ai}(x,y)$$
$$= r_{A1}.P_{A1}(x,y) + r_{A2}.P_{A2}(x,y) + ... + r_{Am}.P_{Am}(x,y)$$

So,

$$P_R(x,y) = \left(\sum_{i=1}^{m} x_{Ai}.P_{Ai}(x,y)\right) - e_B.P_D(x,y)$$

$$= x_{A1}.P_{A1}(x,y) + x_{A2}.P_{A2}(x,y) + ... + x_{Am}.P_{Am}(x,y) - e_B.P_D(x,y)$$

$$= (r_{A1} + e_B.d_{A1}).P_{A1}(x,y) + (r_{A2} + e_B.d_{A2}).P_{A2}(x,y) + ... + (r_{Am} + e_B.d_{Am}).P_{Am}(x,y)$$
$$- e_B.\left(d_{A1}.P_{A1}(x,y) + d_{A2}.P_{A2}(x,y) + ... + d_{Am}.P_{Am}(x,y)\right)$$

$$= r_{A1}.P_{A1}(x,y) + e_B.d_{A1}.P_{A1}(x,y) + r_{A2}.P_{A2}(x,y) + e_B.d_{A2}.P_{A2}(x,y) + ... + r_{Am}.P_{Am}(x,y) + e_B.d_{Am}.P_{Am}(x,y)$$
$$- e_B.d_{A1}.P_{A1}(x,y) - e_B.d_{A2}.P_{A2}(x,y) - ... - e_B.d_{Am}.P_{Am}(x,y)$$

$$= T_A(x,y) + d_{A1}.U_A(x,y) + d_{A2}.V_A(x,y) + ... + d_{Am}.W_A(x,y)$$

$$[T_A(x,y) = r_{A1}.P_{A1}(x,y) + e_B.d_{A1}.P_{A1}(x,y) + r_{A2}.P_{A2}(x,y) + ... + r_{Am}.P_{Am}(x,y)$$

$$U_A(x,y) = -e_B.P_{A1}(x,y), \quad V_A(x,y) = -e_B.P_{A2}(x,y)$$

$$...$$

$$W_A(x,y) = -e_B.P_{Am}(x,y)]$$

$$\Rightarrow P_R(x,y) - T_A(x,y) = d_{A1}.U_A(x,y) + d_{A2}.V_A(x,y) + ... + d_{Am}.W_A(x,y)$$

$$\Rightarrow Z_A(x,y) = d_{A1}.U_A(x,y) + d_{A2}.V_A(x,y) + ... + d_{Am}.W_A(x,y) \quad [Z_A(x,y) = S_A(x,y) - T_A(x,y)]$$

Now, d_{A1}, d_{A2},..., and d_{Am} can't be found out in polynomial time because of the ECDLP of elliptic curves.

Next consider the equation $D_A(x,y) - d_B C_A(x,y) = C_A(x,y)$.
We have,

$$D_A(x,y) - d_B C_A(x,y) = C_A(x,y)$$
$$D_A(x,y) = d_B C_A(x,y) + C_A(x,y)$$
$$= (d_B + 1)C_A(x,y)$$
$$D_A(x,y) = d_B . C_A(x,y) \quad [d_B = d_B + 1]$$

Again, we cannot solve d_B in polynomial time because of the ECDLP of elliptic curves. In the same way, we cannot solve for t_B in the third equation $E_A(x,y) - t_B.G_A(x,y) = 0$ in polynomial time. So, Eve cannot masquerade as Alice.

CONCLUSIONS

Smart cards can add convenience and safety to any transaction of data. But the integration of RSA algorithm in Smart cards is not efficient due to the larger key size demand of RSA to cope up with today's security requirement. ECC may be a right choice for Smart Cards' security. We have also proposed a protocol for Smart card authentication using Elliptic Curves.

REFERENCES

[1] Ian Blake, Gadiel Seroussi, Higel Smart, Elliptic Curves in Cryptography, Cambridge University Press, 1999.
[2] Lawrence C. Washington, Elliptic Curves, Number Theory and Cryptography, CRC Press, 2008.
[3] Joseph H. Silverman, John Tate, Rational Points on Elliptic Curves, Springer, 1992.
[4] Menezes, Okamoto, Vanstone, "Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field, IEEE Transaction on Information Theory, vol. 39, 1993.
[5] Henri Cohen, Gerhard Frey, Handbook of Elliptic and Hyperelliptic Curve Cryptography, CRC Press, 2006.
[6] Neal Koblitz, Alfred J. Menezes, "A survey of public-key cryptosystems,", Aug 7. 2004.
[7] Smart Cards from The Wikipedia website. [Online]. Available: http://en.wikipedia.org/
[8] Joseph H. Silverman, John Tate, Rational Points on Elliptic Curves, Springer, 1992.
[9] Joseph H. Silverman, The Arithmetic of Elliptic Curves, Springer, 1986.
[10] J. J. Shen, C. W. Lin and M. S. Hwang, "A modified remote user authentication scheme using smart cards," IEEE Trans. Consumer Electronic, vol. 49, no. 2, pp. 414-416, May 2003.
[11] Thomas Koshy, Elementary Number Theory with Applications, Academic Press, 2009.
[12] Ian Blake, Gadiel Seroussi, Higel Smart, Advances in Elliptic Curve Cryptography, Cambridge University Press, 2005.
[13] Erdinc Ozturk, "Low Power Elliptic Curve Cryptography" M.Sc thesis, Worcester Polytechnic Institute, April 2004.
[14] Bhattacharya, Jain, Nagpaul, Basic Abstract Algebra, Cambridge University Press, 2002.
[15] Bruice Schneier, Applied Cryptography, Wiley India, 2007.
[16] William Stallings, Cryptography & Network Security, PHI, 2006
[17] Atul Kahate, Cryptography and Network Security, 2E, Tata McGraw, 2011.
[18] Rotman, Galois Theory, Springer International Edition, 2010.
[19] R.L.Rivest, A.Shamir & L.M.Adleman, " A method for obtaining Digital Signature and Public Key Cryptosystems", ACM, 1978.
[20] W. Diffie, P Vanoorschot, and M. Wiener, "Authentication and authenticated key exchanges. Designs, Codes and Cryptography", 107–125, 1992.
[21] Business Security Measures Using SSL, Realtime Publishers.

[22] Kristin Lauter, "The Advantages of Elliptic Curve Cryptography for Wireless Security", Microsoft Corporation.
[23] The Times of India, Kolkata, Tuesday, November 29, 2011, "How to secure your online transactions".
[24] The Accord Fintech, *Wednesday, March 14, 2012,* "State Bank of India likely to install authentication biometric devices".
[25] The Rupee Times, *March 14, 2012,*"SBI to come up with biometric customer authentication devices".

Ranbir Soram is working as a lecturer in Computer Science and Engineering at Manipur Institute of Technology, Takyelpat, Imphal, India. His field of interest includes network security, neural network, genetic algorithm etc.