Mobile SMS Banking Security Using Elliptic Curve Cryptosystem

Ranbir Soram

Manipur Institute of Technology, Takyelpat, Imphal -795004, India

Summary

Over the past many years several leading banks in India have launched SMS banking services. However, the security of mobile SMS payment and banking has topped the list of concerns for most of the customers. In this paper, I investigate the security loopholes in SMS banking and propose a system to make mobile SMS banking secure using Elliptic Curve Cryptosystem(ECC). Also, another aim is to design an API to implement ECC algorithm.

Key words:

SMS, Elliptic Curve, Digital Signature, ECC Banking Module.

1. Introduction

Mobile SMS banking is a term used for performing balance checks, account transactions, payments etc. via a mobile device such as a mobile phone. SMS banking is a new and modern service rendered by most banks in India.The Mobile SMS banking system is based on the exchange of SMS messages between customers and the bank. Mobile banking today is most often performed via SMS. Mobile usage has seen an explosive growth in India. The main reason that Mobile SMS Banking scores over Internet Banking is that it enables 'Anywhere Banking'. Customers now don't need access to a computer terminal to access their banks, they can now do so on the go – when they are waiting for their bus to work, when they are traveling or when they are waiting for their orders to come through in a restaurant. There are two methods of SMS banking services widely used today; they are the push **and pull** SMS messages.

Push SMS message is the message that the bank sends out to a customer's mobile phone, without the customer initiating a request for the information. An example of push message could be a withrawal alert (SBI bank, India), which alerts the user when a withrawal is made from his account.

Pull SMS message is a request initiated by the customer, using a mobile phone, for obtaining information or performing a transaction in the bank account.This is a full duplex communication system where a user sends a request to the bank and the bank replies with the information sought by the user. An example of pull SMS message is an account balance enquiry made by a user.

The other way to categorize the mobile SMS banking services, by the nature of the service, gives us two kinds of services – Enquiry based and Transaction based. So a request for your bank statement is an enquiry based service and a request for your fund transfer to some other account is a transaction based service. Transaction based services are also differentiated from enquiry based services in the sense that they require additional security across the channel from the mobile phone to the banks data servers.

Based upon the above classifications, the following taxonomy of the services listed below can be arrived at.

Table 1: Mobile SMS banking service classification

	Push Based	**Pull Based**
Transaction Based		->Fund Transfer ->Bill Payment ->Other financial service like shared trading
Enquiry Based	>Credit/Debit Alerts. ->Minimum Balance Alerts ->Bill Payment Alert	->Account Balance Enquiry ->Account Statement Enquiry ->Cheque Status Enquiry ->Cheque Book Requests ->Recent transaction history

Manuscript received June 5, 2009
Manuscript revised June 20, 2009

2. The Need for SMS banking

With the rapid growth in the number of mobile phone users in India banks have been exploring the feasibility of using mobile phones as an alternative medium of delivery of banking services. Moreover, the Reserved Bank of India(RBI) has acknowledged Banks as an important partner in offering mobile services in India as it always has issues on someone other than bank collecting money from the public. But due to the nature of connectivity between bank and its customers ,it would be impractical to expect customers to regularly visit banks or connect to the bank's web site for regular business like balance check or account transfer. With mobile SMS banking , users can now conveniently carry out banking transactions 24hrs a day .This is because a user has access to his mobile phone all day, at all times, hence the need for SMS banking.

3. How SMS works

The Short Message Service (SMS) is a GSM technology that allows exchange of text messages up to 160 characters among mobile phones through the short message service Center (SMSC) of the particular network operator. The relative ease of the use of SMS makes it the most wanted means of communication among mobile users.

SMS uses the GSM special signaling channel instead of voice channel and is therefore a very cheap and reliable media channel. The SMSC receives the message from the mobile device and routes it to the destination device. The operation of SMSC is offered as a service by many service providers. The SMS processing computers usually run on corporate servers that are connected to the SMS network through specialized routers and gateways connected to the SMS centers of the mobile operators. These servers are assigned short numbers usually 3 to 6 digits long. These numbers are operator specific.

4. Security Problems with SMS

The initial idea for SMS usage was intended for the subscribers to send non-sensitive text messages across the open GSM network. Mutual authentications, text encryptions, end-to-end security, non-repudiation and many security concerns for SMS were omitted during the design of GSM architecture. The only encryption involved during transmission is the encryption between the base transceiver station and the mobile station. End-to-end encryption is currently not available. The encryption algorithm used is A5 which is proven to be vulnerable. Therefore, a more secure algorithm is needed.

5. Analysis of the Traditional Mobile SMS Banking System

In compliance with the directive of the Reserved Bank of India, presently, customers have to walk in to the bank to submit the registration form by giving their mobile number, account number and transaction details. The presence of the customer in the bank is required. Each customer is given a 4-digit number for authentication (may vary bank to bank) . The customer can receive his account balance and transactions only when the request is received from the mobile number registered with the bank and duly authenticated by the 4-digit number. The mobile number and the 4-digit number serve as a User ID and password for authentication. The 4-digit code number has therefore to be kept confidential. Data carried across the mobile network are protected by the standard GSM security protocols at the communication layer. The subscriber identity is also protected across this chain. But the risk in transporting data across the GSM channel may be found in the number of stops the data make before reaching the bank as shown in the block diagram in figure 2. A customer would initiate a transaction by sending SMS to the bank using the bank's SMS short code as a terminating address. The SMS would be automatically stored on the handset and be available to anyone that looks at the customer's phone. The SMS would then pass through the encrypted GSM channel, through the base stations and terminate at the mobile network operator, where it is typically stored unencrypted. Unlike the fix line communication, data being carried across the mobile network jumps from one base station to the next, which means that the chain of encrypted communication between the customer and the bank is broken. As can be seen , there are many points of exposure. So, current mobile banking services offered by banks are not secure enough to protect confidential data.

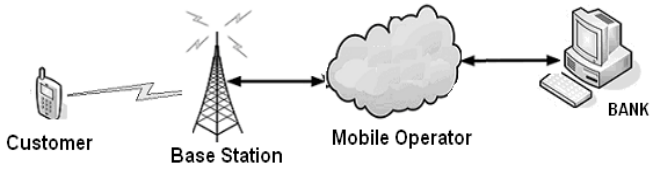

Figure 2: Mobile SMS Banking Architecture

6. The Proposed System-ECC Banking Module

After a thorough study of the above traditional system, a system that would provide security at the satisfaction of the customers is proposed. This system is called the **ECC Banking module.** This ECC Banking

module receives the text messages from the clients/banks and processes them and sends the output back to the banks/users as and when required. This ECC Banking module provides secure data encryption and decryption using public key cryptography .The technology is perfectly secure and GPRS is not mandatory. Not many phone users in India subscribes to GPRS and even fewer have phones that can support GPRS. Around 70 percent of the 350 million handset connections in India are without GPRS . Due to lack of GPRS connectivity, secure SMS based applications will be prominent in the mobile banking. The block diagram for the new system is given in figure 3.

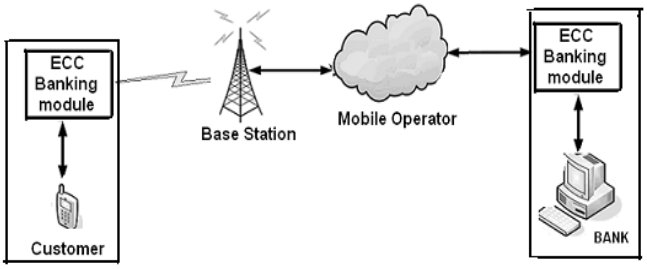

Figure 3:Proposed Mobile SMS Banking Architecture

The silent features of the new system are enumerated below:-

(i) A strong cryptographic technology called the Elliptic Curve Cryptography, instead of traditional RSA, is used. We use public and private keys for encryption /decryption purpose. A quick discussion on Elliptic Curve Cryptography is given in the next section.

(ii) Two ECC Banking modules, one in the handset and another in the bank, are used.

(iii) A Digital signature using ECC technology may optionally be used in each module to provide message authentication, message integrity and non-repudiation. Mobile SMS banking in India does not provide these.

The ECC Banking module in the handset is one of the design issues in this very proposed system and is addressed here. One solution proposed in this paper is to house the ECC Banking module as a *SIM Application Toolkit(SAT)* . The SAT is standard of the GSM system which enables the SIM to initiate actions which can be used for various services. The SAT consists of a set of commands programmed into the SIM card, which defines how the SIM should interact directly with the outside world and initiates commands independently of the handset and the network. This enables the SIM to build up an interactive exchange between a network application and the end user, and access or control access to the network. The STK has been deployed by many mobile operators around the world for many applications. The challenge in SIM based applications is getting the application onto a SIM card that already exists in the market. The service provider has the option of sending the application Over The Air (OTA), which entails the delivery of several encrypted SMS messages that self-configure the application on the SIM, or, provisioning a new SIM card with the application already embedded within the SIM. The latter has an economic impact on the network operator and the existing customer would have to obtain a new SIM card in order to use the application. Once the ECC Banking module is on the SIM, message from the customers can be encrypted, and transported by SMS to the service provider or bank. A benefit of SIM Based Applications is the ability of the network operator or bank to own a piece of the real estate on the SIM Card.

Another option to house the ECC Banking module is in the handset itself. New handset can come with pre-built ECC Banking module. Existing handsets may be reprogrammed to house the ECC Banking module.

7. Cryptography with Elliptic Curves

For those unfamiliar with the basic theory of elliptic curves, this paper discusses a brief introduction to this field; more complete introductions appear in, for example, Silverman [1986], Koblitz[1987], and Tibor Juhas[2007].

In 1985 Victor Miller, who was then at IBM, and Neil Koblitz from the university of Washington first introduced the Elliptic Curve Public Key Cryptography system, a method based on the Discrete Logarithmic problem over the points on an elliptic curve. The principal attraction of ECC compared to RSA is that it offers equal security for a far smaller key size, thereby reducing processing overhead. The ECC has received considerable attention from mathematicians around the world, and no significant breakthroughs have been made in weaknesses in the algorithm.

An elliptic curve is the set of equations of the form

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

or

$$y^2 + xy = x^3 + ax^2 + b \quad (2$$

or

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

where **x** and **y** are variables, **a** and **b** are constants. However, these values are not necessarily real numbers; instead they may be values from any field. For cryptography purposes we always use a finite field.

8. Elliptic curve over a Galois field

Using the real numbers for cryptography have 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. 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, p^n and is denoted by $GF(p^n)$. Two special Galois fields are standard for use in Elliptic Curve cryptography. They are GF(p) when n=1 and $GF(2^n)$ when p=2

9. Elliptic curve over a GF(p)

Elliptic curves over GF(p) are of the form

$E_p(a,b)$: $y^2 \pmod p = x^3+ax+b \bmod p$

where $a,b \in F_p$ and $(4a^3+27b^2) \bmod p \neq 0$.

All operations such as addition, subtraction, division, multiplication involves integers between **0** to **p**-1. The prime **p** is chosen such that there is finitely large number of points on the elliptic curve to make cryptosystem secure. SEC specifies elliptic curve with **p** ranging between 112-521 bits.

The addition of two points $P(x_1,y_1)$ and $Q(x_2,y_2)$ is calculated as

$$R(x_3,y_3)=P+Q$$
$$x_3=\lambda^2-x_1-x_2$$
$$y_3=\lambda(x_1-x_3)-y_1$$
$$\lambda=(y_2-y_1)/(x_2-x_1) \text{ if } P\neq Q$$
$$\lambda=(3x_1^2+a)/2y_1 \quad \text{if } P=Q$$

The multiplication of points by a scalar is a series of additions and doubling of points. That is, we define ***kP***, where ***k*** an integer and ***P*** is a point , by repeated addition in the natural manner. The value of *kP* may be efficiently computed via repeated doubling . That is,

$$kP = \begin{cases} P \text{ for } k=1 \\ (P+P) \times k/2 \quad \text{for k even} \\ P+(k-1) \times P \quad \text{for k odd} \end{cases}$$

The multiplication by –1 converts **P** to **–P** by negating the **y** coordinate of **P** i,e the negative of **P**=(x,y) gives **–P**=(x,**-y**).

10. Finding points on the curve

The following algorithm in figure 4 gives the points on the curve $E_p(a,b)$.

```
Algorithm elliptic_points(p,a,b)
{
x=0
while(x<p)
{
w=(x^3+ax+b) mod p
if( w is a perfect square in Z_p )
  output ((x,√w),(x,-√w))
x=x+1
}
}
```

Figure 4. Algorithm to find points on the curve $E_p(a,b)$

It is of some interest to know the number of points in a finite abelian group defined over an elliptic curve. The number of points on an elliptic curve over a finite field must satisfy Hasse's theorem. Given a field GF(p), the order of the curve N will satisfy the following equation.

$$P+1-2\sqrt{P} \leq N \leq P+1+2\sqrt{P} \quad (4)$$

Note that the number of points in $E_p(a,b)$ is approximately equal to the number of elements in Z_p ,namely **P** elements.

11. Elliptic curve over a $GF(2^n)$

Let us look at the elliptic curve over $GF(2^n)$.That means our constants are either polynomial or normal basis numbers. It also means we cannot use the elliptic equation which we used for real numbers. The mathematicians tell us that we need to use either this version

$$y^2+xy = x^3+ax^2+b \quad (5)$$

or this version

$$y^2+y = x^3+ax+b \quad (6)$$

Note that the value of **x**,**y**,**a**, and **b** are polynomials representing **n**-bit words.

Mathematicians say that the second form above, equation (6), 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 (5) are called "nonsupersingular". Till 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 cryptanalysis to attack.

For equation (5) to be valid, ***b*** must never be zero. However, ***a*** can be zero.

The rules for adding two points in GF(2^n) is slightly different from the rules for GF(p).

1. If $P=(x_1,y_1)$, $Q=(x_2,y_2)$**,** $-P\neq Q$, $P\neq Q$, then $R(x_3,y_3)=P+Q$ can be computed as
$$\lambda=(y_2+y_1)/(x_2+x_1)$$
$$x_3=\lambda^2+\lambda+x_1+x_2+a$$
$$y_3=\lambda(x_1+x_3)+ x_3+ y_1$$

2. If P=Q, then $R(x_3,y_3)=P+P$ (or R=2P) can be computed as
$$\lambda=y_1/x_1+x_1$$
$$x_3=\lambda^2+\lambda+a$$
$$y_3=(\lambda+1)x_3+ x_1^2$$

The multiplication by –1 converts **P** to **–P** by adding x to the **y** coordinate of **P** i,e the negative of **P**=(x,y) gives **–P**=(x,x+**y**).

12. ECC Encryption/Decryption

Several methods have been used to encrypt and decrypt using elliptic curves. This paper describes a method based on the simulation of the EIGamel cryptosystem using an elliptic curve over GF(p). The following are the steps involved in the process.

12.1 Generating public and private keys

1. Bob chooses E(a,b) with an elliptic curve over GF(p).
2. Bob chooses a generator point, $e_1(x_1,y_1)$ on the curve.
3. Bob chooses an integer **d**.
4. Bob calculates $e_2(x_2,y_2)=dxe_1(x_1,y_1)$. Multiplication here means multiple additions of points.
5. Bob announces $E(a,b)$,$e_1(x_1,y_1)$ and $e_2(x_2,y_2)$ as his public key; he keeps **d** as his private key.

12.2 Representation of message as elliptic curve point

Let us assume that s_i is an integer in F_q and $s_i=x^3+ax+b$ mod p. The probability that $s_i=x^3+ax+b$ mod p is a square mod p is ½. Let **r** is an integer so that a failure rate $1/2^{\mathbf{r}}$ is acceptable when trying to encode a message as a point.

The method can be defined as follows:

1. Express the message as number **m**, m≥0, such that **m** x **r** < **p**.
2. Assume $x_i=mxr+i$ for i Є [0,r] and compute $s_i=x_i^3+ax_i+b$ until we get $s_i^{(p-1)/2}=1$ mod p.
3. Compute the square root modulo p of s_i as y_i.
4. The point $P=(x_i,y_i)$ is the representation of the message.

12.3 Encryption

Alice selects **P**, a point on the curve, as her plaintext, **P**. She then calculates a pair of points on the text as ciphertexts:

$C_1=r\mathbf{x}e_1$ $C_2=\mathbf{P}+r\ \mathbf{x}\ e_2$

We may wonder how an arbitrary plaintext can be a point on the elliptic curve. This is one of the challenging issues in the use of the elliptic curve for simulation. Alice needs to use an algorithm to find a one-to-one correspondence between a block of text and the points on the curve.

12.4 Decryption

Bob, after receiving $\mathbf{C_1}$ and $\mathbf{C_2}$, calculates **P**, the plaintext using the following formula, $P=C_2-(d\ x\ C_1)$. The minus sign here means adding the inverse.

We can prove that the **P** calculated by Bob is the same as that sent by Alice, as shown below:

$$\mathbf{P+r\ x\ e_2-(d\ x\ r\ x\ e_1)=P+(r\ x\ d\ x\ e_1)-(r\ x\ d\ x\ e_1)=P+O=P.}$$

P, C1, C2, e1, e2 are all points on the curve. Note the result of adding two inverse points on the curve is the **zero** point.

12.5 Representation of elliptic curve point as message

1. Compute $m=[x_i/r]$.
2. Convert **m** into message.

12.6 ECC Example

To illustrate the ECC, let us consider the following elliptic curve:

$$y^2=x^3-x+188 \bmod 751 \qquad (7)$$

The elliptic curve group generated by the above elliptic curve is $E_{751}(-1,188)$.

Let the generator point $\mathbf{e_1}=(0,376)$. Then multiples of $\mathbf{dxe_1}$ of the generator point $\mathbf{e_1}$ are (for $1\leq\mathbf{d}\leq751$):

$\mathbf{e_1}=(0,376)$ $2\mathbf{e_1}=(1,376)$ $3\mathbf{e_1}=(750,375)$

$4\mathbf{e_1}$=(2,373) $\mathbf{5e_1}$=(188,657) $6\mathbf{e_1}$=(6,390)
$7\mathbf{e_1}$=(667,571) $8\mathbf{e_1}$=(121,39) $9\mathbf{e_1}$=(582,736)
$10\mathbf{e_1}$=(57,332) …… $761\mathbf{e_1}$=(565,312) $762\mathbf{e_1}$=(328,569)
$763\mathbf{e_1}$=(677,185) $764\mathbf{e_1}$=(196,681) $765\mathbf{e_1}$=(417,320)
$766\mathbf{e_1}$=(3,370) $767\mathbf{e_1}$=(1,377)
$768\mathbf{e_1}$=(0,375) $769\mathbf{e_1}$=O (point at infinity)

If Alice wants to send to Bob the message M which is encoded as the plaintext point P=(443,253)$\in E_{751}(-1,188)$, she must use Bob public key to encrypt it. Suppose that Bob secret key **d=85,** then his public key will be $e_2=dxe_1=85(0,376)$ =(671,558).

Alice selects a random number **r**=113 and uses Bob's public key e2=(671,558) to encrypt the message point into the ciphertext pair of points:

$[C1,C2]=[(re_1),(P+rxe_2)]$
$=[113x(0,376),(443,253)+113x(671,558)]$
$=[(34,633),(443,253)+(47,416)]$
$=[(34,633),(217,606)]$

Upon receiving the ciphertext pair of points, [C1,C2] =[(34,633),(217,606)],Bob uses his private key, **d**=85, to compute the plaintext point, **P**, as follows

$(P+r \times e_2)-[d \times (r \times e_1)]=(217,606)-[85x(34,633)]$
$=(217,606)-[(47,416)]$
$=(217,606)+[(47,-416)]$ since $-P=(x_1,-y_1)$
$=(217,606)+[(47,335)]$ since $-416 \equiv 335 \bmod 751$
=(443,253)
and then maps the plaintext point **P**=(443,253) back into the original plaintext message **M**.

13. Digital Signature

We are all familiar with the concept of a signature. We sign a document to show that it originated from us or was approved by us. The signature is the proof to the recipient that the document comes from the correct sender.

A conventional signature is included in the document itself; it is a part of the document. When we sign a cheque, the signature is on the cheque; it is not a separate document; on the other hand, when we sign a document digitally, we send the signature as a separate document. The sender sends two documents; the message and the signature. The recipient receives both documents and verifies that signature is of the sender.

It is to be noted there is a distinction between private and public keys as used in digital signature and public and private keys as used in cryptosystem. In a cryptosystem, we use the private and public keys of the recipient; in a digital signature, we use the private and public keys of the sender.

I consider here the elliptic curve digital signature scheme, which is a digital signature algorithm (DSA) based on elliptic curves. This scheme sometimes is referred to as ECDSA (elliptic curve DSA). The ECDSA consists of the following three steps:

1) Key generation
2) Signing
3) Verifying

13.1 Key Generation

Key generation follows these steps:

1. Alice chooses and elliptic curve $E_p(\boldsymbol{a,b})$ with $\boldsymbol{p}$ a prime number.
2. Alice chooses another prime number $\boldsymbol{q}$ to be used in the calculation.
3. Alice chooses an integer $\boldsymbol{d}$ as the private key.
4. Alice chooses a point $\boldsymbol{e_1}$ on the curve.
5. Alice calculates $\boldsymbol{e_2} = \boldsymbol{d} \times \boldsymbol{e_1}$ as another point on the curve.
6. Alice's public key is ($\boldsymbol{a, b, p, q, e_1, e_2}$) and her private key is $\boldsymbol{d}$.

13.2 Signing

The signing process consists mainly of choosing a secret random number, creating a third point on the curve, calculating two signatures. And sending the message and the signatures.

1. Alice chooses a secret random number $\boldsymbol{r}$ between $\boldsymbol{1}$ and $\boldsymbol{q-1}$.
2. Alice selects a third point on the curve, $P(u,v)=\mathbf{r}x\boldsymbol{e_1}$.
3. Alice uses the first coordinates of P(u,v) to calculate the first signature $\mathbf{S_1}$. This means $\mathbf{S_1}=\mathbf{u} \bmod \boldsymbol{q}$.
4. Alice uses the digest of the message, her private key, and the secret random number $\boldsymbol{r}$, and the $\mathbf{S_1}$ to calculate the second signature $S_2=(h(M)+dxS_1)r^{-1} \bmod \boldsymbol{q}$.
5. Alice sends M, S_1, S_2.

13.3 Verifying

The verification process consists mainly of-reconstructing the third point and verifying that the first coordinate is equivalent to $\mathbf{S_1}$ in modulo **q**. The third point was created by the signer using the secret random number r. The verifier does not have this value. He needs to make the third point from the message digest, $\mathbf{S_1}$ and $\mathbf{S_2}$.

1. Bob uses M, S_1, and S_2 to create two intermediate results, A and B:
$A = h(M)\, S_2^{-1} \bmod q$ and $B = S_2^{-1}\, S_1 \bmod q$.

Bob then reconstructs the third point

$$T(x,y) = A \times e_1 + B \times e_2.$$

2. Bob uses the first coordinate of T(x, y) to verify the message. If $x = S_1 \bmod \mathbf{q}$, the signature is verified; otherwise; it is rejected.

14. Development of API

There is the question about where the encryption gear should be located. There are two approaches: ***link encryption*** and ***end-to-end encryption.*** With link encryption, each vulnerable communications link is equipped on both ends with an encryption device. Thus, all traffic over all communication links is secured. One of its disadvantages is that the message must be decrypted each time it enters a switch because the switch must read the address in the message header in order to route the data. Thus, the message is exposed at each switch. This is undesirable in our Mobile SMS banking.

With end-to-end encryption, the encryption process is carried out at the two end systems. The data in encrypted form are transmitted unaltered across the network to the destination terminal. Thus the end-to-end encryption relieves the customers of concerns about the degree of security of networks. Thus, end-to-end encryption makes user data secure.

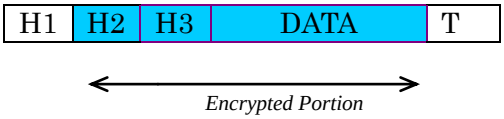

Figure 5. Network Layer Encryption. In this case, data and more headers are encrypted. Here **H1**, **H2** and **H3** are headers and **T** is trailer.

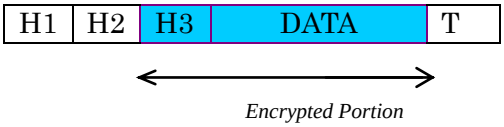

Figure 6. Transport Layer Encryption. In this case, data and less headers are encrypted. Here **H1**, **H2** and **H3** are headers and **T** is trailer.

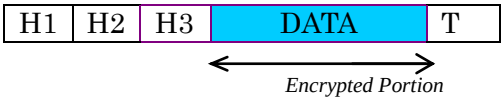

Figure 7. Application Layer Encryption. In this case, only data portion is encrypted. Here **H1**, **H2** and **H3** are headers and **T** is trailer.

An interesting way of viewing the encryption alternatives as given in the above figures is that as we move up to the API layers, less information is encrypted but it is more secure.

For end-to-end encryption, several choices are possible for the placement of the encryption function. We can place it in the network layer or transport layer. The user data portion and some headers of all frames are encrypted. See figures 5 and 6 given above. However, if the message passes through a node or gateway, the line connection is terminated and a new connection is opened for the next hop or node. Thus, encrypted data (our message + some headers) are decrypted at the gateway. Before transmission to the next node, it is again encrypted. So, our data are not secure at the intermediate nodes like gateways.

For application that has a store-and-forward capacity, the only place to achieve end-to-end encryption is at the application layer. A drawback of application layer encryption is that the number of entities increases considerably. A network that supports hundreds of hosts may support many users and processes. So, many more secret keys need to be generated and distributed.

With application level encryption, only the user data portion of a segment is encrypted. See figure 7 given above. The headers are all clear and visible. So, no decryption and encryption of data take place at the intermediate nodes or gateways. Thus, our data are still secure at the intermediate nodes. For this reason it is desirable to create an API that implements encryption and decryption at the application layer. As the model suggests, the ECC API is intended to be used in the application layer to automatically encrypt/decrypt all data that flows to or from the lower layers.

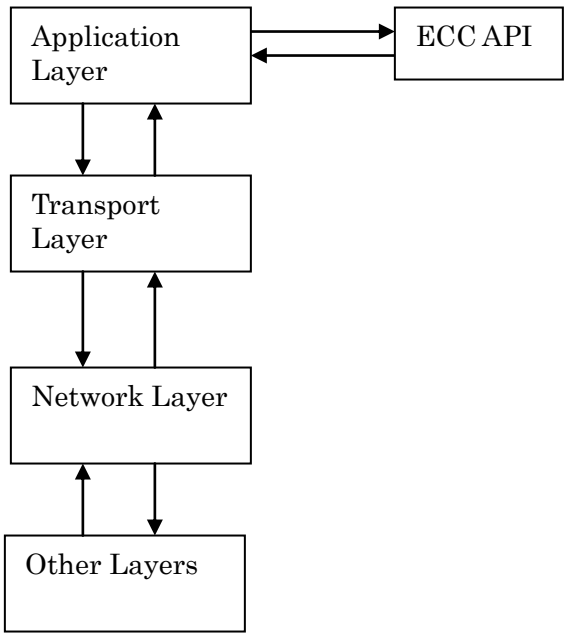

Figure 8: Where the API fits in the overall network

15. Security of ECC

The security of ECC depends on the difficulty of Elliptic Curve Discrete Logarithmic Problem. Let **P** and **Q** be any two points on an elliptic curve such that k**P**=**Q**, where **k** is a scalar. Then the Elliptic Curve Discrete Logarithmic Problem is to determine **k** given **P** and **Q**. It is relatively easy to calculate **Q** given **k** and **P**,but it is very hard to determine **k** given **Q** and **P**. As an example consider the group $E_{23}(9,17)$,defined by the elliptic curve equation $y^2 \pmod{23} = (x^3+9x+17) \bmod 23$. What is the discrete logarithm **k** of **Q**=(4,5) to the base **P**=(16,5)? The brute-force method is to compute multiples of **P** until **Q** is found. Thus,

P=(16,5); 2P=(20,20);3P=(14,14); 4P=(19,20); 5P=(13,10) ; 6P=(7,3) ;7P=(8,7) ; 9P=(4,5).

Because 9**P**=(4,5)=**Q**, the discrete logarithm of Q=(4,5) to the base P=(16,5) is **k**=9. In a real application, **k** would be so large as to make the brute-force method infeasible.

16. Easier calculation

Another aspect which distinguishes elliptic curve cryptosystem from other cryptosystem (say RSA) is the easier calculations required for producing the cryptographic parameters. Therefore, elliptic curve cryptosystem better fits for implementations on devices with reduced resources such as mobile phones.

17. Advantages and Limitations of Mobile SMS banking.

17.1 Advantages:

(i) Mobile SMS Banking can be done from any GSM handset as all GSM handsets support SMS.

(ii) Mobile SMS Banking saves a lot of customers' time as they need not go to banks for enjoying transactions.

(iii) Relatively, Mobile SMS Banking reduces costs as the cost of sending an SMS is very cheap.

(iv) Mobile SMS Banking makes a lot of conveniences to the customers as they can perform transaction anywhere and anytime.

17.2 Limitations:

(i) An SMS message may consist of a maximum of 160 characters. This is one of the limitations in Mobile SMS Banking.

(ii) The SMS is a store-and-forward based system. So, it does not guarantee delivery of messages.

(iii) Some locations may not have network coverage resulting in the unavailability of Mobile SMS Banking

(iv) Most SIM cards have limited space (about 32Kb) hence low function.

18. Conclusion

This paper studied Mobile SMS banking security by means of elliptic curve cryptographic technique. Approaches based solely on data encryption provide a security that depends on the place encryption/decryption is actually performed. As the security of the proposed system is very hard , it is very clear that the proposed Mobile SMS Banking will dominate banking sector in India. It has been mentioned in many literatures that a considerably smaller key size can be used for ECC compared to RSA. Also mathematical calculations required by elliptic curve cryptosystem are easier, hence, require a low calculation power. Therefore ECC is a more appropriate cryptosystem to be used on small devices like mobile phones.

I have also developed an API that implements ECC, allowing it to be used in banking sectors.

19. Future Work

In the course of this work, I identified areas that are needed to carry out our future work. Area like the placement of the ECC Banking module in the SIMs or handsets needs future work for proper implementation.

Acknowledgment

The author would like to express his sincere thanks to his daughter, ***Java Compiler Soram*** and son, ***Chandrayan One Soram*** for not disturbing him at his work.

References

[1] Silverman,The arithmetic of elliptic curves, In Graduate Texts in Mathematics, vol.106,1986

[2] N. Koblitz, Elliptic Curve Cryptosystems, Mathematics of computation, No.48,1987,203-209.

[3] Tibor Juhas,The Use of Elliptic Curves in Cryptography, MasterThesis,Universityof Tromso, 2007.

[4] T. EIGamal,A public key cryptosystem and a signature scheme based on discrete logarithms,IEEE Transactions on Information Theory, Vol.31,1985

[5] N.P. Smart,The Discrete logarithm Problem on Elliptic curves of Trace One, Journal of Cryptology, Vol.12,1999
[6] Stalling W,Cryptography and Network Security,Prentice-Hall, 2001.
[7] Kumanduri and Romero, Number Theory with Computer Applications, Prentice- Hall, 2001.
[8] A. Lenstra and E. Verheul, Selecting Cryptographic Key Sizes, Journal of Cryptology,14 ,2001,255-293.
[9] IEEE P1363, Standard Specifications for Public-Key Cryptography, Institute of Electrical and Electronics Engineers, 2000.
[10] Blake,Seroussi, and Smart, Elliptic Curves in Cryptography,Cambridge University Press,1999.

Ranbir Soram received his B.E. degree in Computer Science and Engineering from Bharathiar University, Coimbatore, India in 1999. Presently he is working as a lecturer in Computer Science and Engineering at Manipur Institute of Technology,Imphal, India. His field of interest includes network security, neural network, genetic algorithm etc.