ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 8, August 2013

Design and Performance Analysis of Secure Elliptic Curve Cryptosystem

Gururaja.H.S.[1], M.Seetha[2], Anjan.K.Koundinya[3]

Assistant Professor, Department of Information Science & Engineering, B.M.S. College of Engineering, Bangalore, India [1]

Professor, Department of Computer Science & Engineering, G.Narayanamma Institute of Technology and Science, Hyderabad, India [2]

Assistant Professor, Department of Computer Science & Engineering, R.V. College of Engineering, Bangalore, India [3]

Abstract: In public-key cryptosystems, the use of RSA and Diffie-Hellman cryptosystems are not adequate due to the use of large number of bits. Elliptic Curve Cryptography has become one of the latest trend in the field of public-key cryptography. Even though, Elliptic Curve Cryptography promises a faster and more secure method of encryption compared to any other standard public-key cryptosystem, there are possibilities of making the algorithm more efficient and secure. This paper illustrates a new design of Elliptic Curve Cryptography implementation which makes it more infeasible attempting any subliminal attack to break the cryptosystem.

Keywords: Elliptic Curve Cryptography, Subliminal Channel, Public-Key Cryptosystem, Encryption, Decryption, Key Generation, Random Number Generation

I. INTRODUCTION

Today, the world is growing fast with electronic revolution in our day-to-day life. In this environment, any information is availab.le at the click of a mouse button. When information is available so freely in the web, there are people who might use this information against us. So it has become mandatory to check this behaviour and control the act of attack against us. This made people to think about a solution named Encryption. The process of Encryption is to convert your information into a form which cannot be read or understood by anyone.

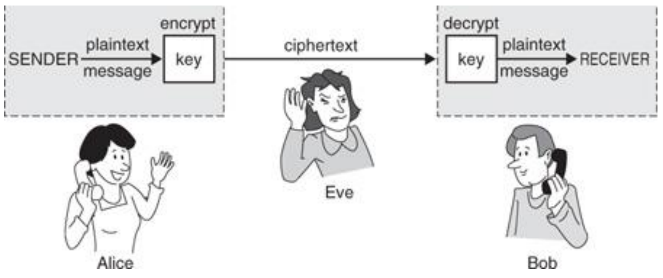

Fig. 1. Process of Cryptography

Fig. 1 shows the process of Cryptography which is the science of secrect sharing. Any character or alphabet we press on the keyboard is represented as a binary information. If the entire document is converted into binary information, just imagine how difficult it would be to recognize the information. In the process of encryption, we try to mix these binary numbers after performing a few mathematical operations which converts the huge set of binary information in a newer arrangement which is difficult to understand.

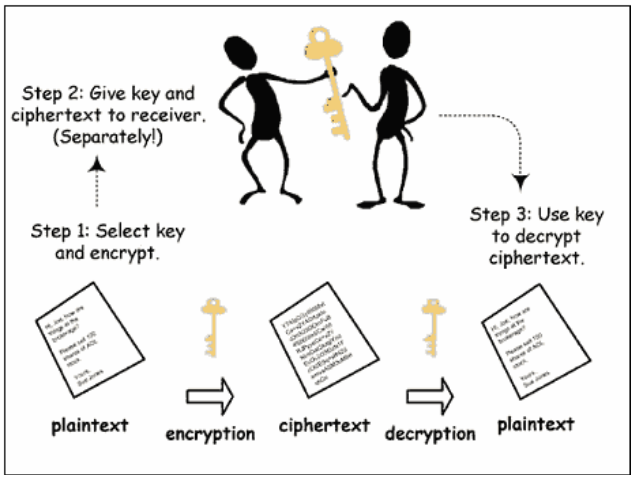
Step 2: Give key and ciphertext to receiver. (Separately!)
Step 1: Select key and encrypt.
Step 3: Use key to decrypt ciphertext.
plaintext
encryption
ciphertext
decryption
plaintext

Fig. 2. Symmetric Key Encryption

The entire process of encryption is to perform mathematical operations on the text that scrambles the text binaries. This is done with the help of a key. Key acts as a parameter which is passed in the encryption function along with the text which needs to be encrypted. The process of encryption then generates a Ciphertext which is nothing but a coded information difficult for others to understand. This can be mathematically represented as

Ciphertext = Encryption (Key, Plaintext) **(1)**

Now the Ciphertext can be transmitted to any unsecure network. When it is received by the intended recepient, the Ciphertext is decrypted using a Decryption function and Key. This generates back the Plaintext. This can be mathematically represented as

Plaintext = Decryption (Key, Ciphertext) **(2)**

Fig. 2 shows the process of Symmetric Key Encryption involving the use of a single key. The equations (1) and (2) gives us the method of generating Ciphertext and Plaintext in Symmetric Key Encryption.

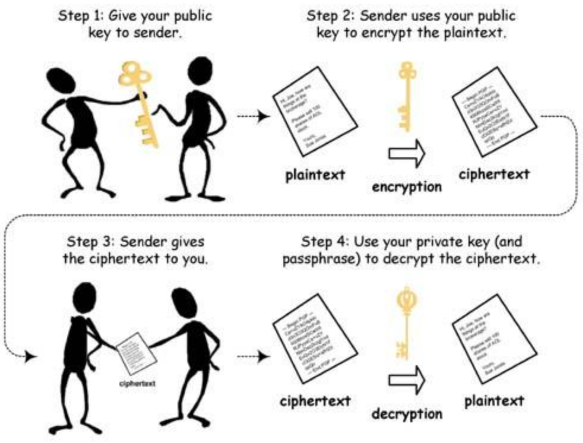
Step 1: Give your public key to sender.
Step 2: Sender uses your public key to encrypt the plaintext.
plaintext
encryption
ciphertext
Step 3: Sender gives the ciphertext to you.
Step 4: Use your private key (and passphrase) to decrypt the ciphertext.
ciphertext
ciphertext
decryption
plaintext

Fig. 3. Asymmetric/Public Key Encryption

Fig. 3 shows the process of Asymmetric Key Encryption or Public Key Encryption which involves the use of two distict keys – one to encrypt and a different one to decrypt.

In public-key cryptosystems, the use of RSA [1,2] and Diffie-Hellman Cryptosystems [3] are not adequate due to the use of large number of bits. This paper mainly focuses on Elliptic Curve Cryptography (ECC) which has a very unique property which makes it suitable for use in Cryptography. Even though, Elliptic Curve Cryptography promises a faster and more secure method of encryption compared to any other standard public-key cryptosystem, there are possibilities of making the algorithm more efficient and secure. This paper illustrates a new design of Elliptic Curve Cryptography implementation which makes it more infeasible attempting any attack in Subliminal Channels [4] to break the cryptosystem.

II. Implementation of Standard ECC

Public Key Cryptosystems are based on the idea of creation of mathematical puzzles which are difficult to solve without the prime knowledge of how they were actually created. The sender keeps the knowledge secret (called Private Key) and only publishes the puzzle (called Public Key). The puzzle can then be used to scramble a message in such a way that only the sender can unscramble.

Early Public Key Cryptosystems like RSA used product of two large prime numbers and publish their product as puzzles (the Public Key). The difficulty in factoring ensures that no one else can derive the private key from the public key. In recent years, due to progress in factoring, public keys in RSA should be thousands of bits in length to provide adequate security.

Elliptic Curve Cryptography is based on the algebraic structure of elliptic curve over finite fields. The class of puzzle used in ECC is slightly different. The puzzle involves the equation $a^b = c$ for 'b' when 'a' and 'c' are known. The equations involving real or complex numbers can be easily solved with the help of logarithms. However, in a large finite group, finding solutions to such equations is quite difficult and is known as discrete logarithm problem [5,6,7,8]. ECC is also used in several integer factorization algorithms that have applications in cryptography.

The equation of an elliptic curve is given as

$$y^2 = x^3 + ax + b \tag{3}$$

Fig. 4 shows an example of a simple elliptic cuve for equation (3).

The following terms are used in the below context:

E - Elliptic Curve

G - Point on the curve

n - Maximum limit (should be a prime number)

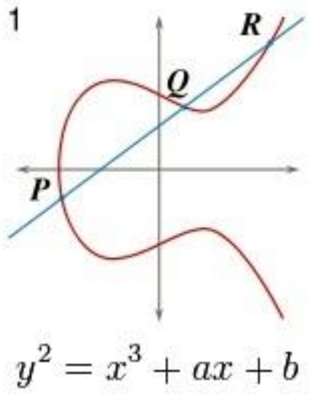

Fig. 4. Example of a Simple Elliptic Curve

In the process of cryptography, key generation is an important part where we have to generate both the public key and private key. The sender will be encrypting the message with receiver's public key and the receiver will decrypt the message with the private key. Fig. 5 shows the process of key generation in Standard ECC where we select a number 'd' within the range of 'n' and generate the public key using the equation Q = d * G where 'd' is the private key and 'G' is a point on the curve.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 8, August 2013

Algorithm Key Generation

```
procedure KEY_PAIR_GEN
Select a number 'd' within the range of 'n'.
Generate the public key using the equation $Q = d * G$
Where d is the random number that we have selected within the range of 1 to (n-1).
G is the point on the curve.
Q is the public key and 'd' is the private key.
```

Fig. 5. Key Generation Algorithm in Standard ECC

Fig. 6 shows the process of encryption in Standard ECC. Let 'm' be the message that we are sending. We have to represent this message on the curve. This has in-depth implementation details. Here we consider 'm' having point 'M' on the curve 'E'. We have to randomly select 'k' in the interval of 1 to (n-1). The ciphertext will be generated using the equation $C = m^Q \bmod q$ where 'q' is the prime number in the interval 1 to (n-1).

Algorithm Encryption

```
procedure ENCRYPT_MSG
Consider 'm' has the point 'M' on the curve 'E'.
Randomly select 'k' in the interval 1 to (n-1).
'q' is the prime number in the interval 1 to (n-1).
Cipher text C will be generated using the equation
$C = m^Q \bmod q$
Ciphertext C will be sent.
```

Fig. 6. Encryption Algorithm in Standard ECC

Fig. 7 shows the process of decryption in Standard ECC. Let 'm' be the message we want to receive. The plaintext will be generated by using the equation $M = C^d \bmod q$ where 'd' is the private key and 'q' is the prime number in the interval 1 to (n-1). As we saw in key generation, 'd' is a selected prime number and can be easily obtained by permutation.

Algorithm Decryption

```
procedure DECRYPT_MSG
Plaintext M will be generated using the equation
$M = C^d \bmod q$
Where M is the original message.
'q' is the prime number in the interval 1 to (n-1).
'd' is the private key.
```

Fig. 7. Decryption Algorithm in Standard ECC

III. Design of Implemented ECC

In public-key cryptosystems, the use of RSA and Diffie-Hellman cryptosystems are not adequate due to the use of large number of bits. Elliptic Curve Cryptography has become one of the latest trend in the field of public-key cryptography. Even though, Elliptic Curve Cryptography promises a faster and more secure method of encryption compared to any other standard public-key cryptosystem, there are possibilities of making the algorithm more efficient and secure. The changes made in the encryption, decryption and key generation algorithms illustrates a new design of Elliptic Curve Cryptography implementation which makes it more infeasible attempting any subliminal attack to break the cryptosystem.

Algorithm Key Generation

```
procedure KEY_PAIR_GEN
Agree on Domain Parameters for the Elliptic Curve E given by $y^2 = x^3 + ax + b$
p is the prime specifying the base field.
A and B are Curve Constants
G = (x,y) is the base point i.e. a point in E of prime order, with x and y being its x- and y- coordinates respectively.
Generate the Private Key X = RANDOM_NUM_GEN
Generate the Public Key $Y = X * G$
This multiplication is achieved using successive Point Doubling and Addition
```

Fig. 8. Key Generation Algorithm in Implemented ECC

In Standard ECC key generation, we select a number 'd' within the range of 'n' and generate the public key using the equation $Q = d * G$ where 'd' is the private key and 'G' is a point on the curve. The problem pertaining to the effectiveness of the process depends on the selection of the private key. If private key is generated only by selecting a number in the range of 'n', it becomes easy for the attacker to predict the private key thus decrypting the plaintext message very easily. Instead, the idea of Random Number Generation [9,10] could be used with the help of various Random Number Generator algorithms to select the number 'd' (private key) which makes it hard for the attacker to predict the private key thus making the ECC algorithm more efficient and secure. Fig. 8 shows the new key generation algorithm for Implemented ECC.

Algorithm Encryption

```
procedure ENCRYPT_MSG
Convert the message string into an array of characters.
Encrypt each character $A_i$ as $C_i = (A_i)^Y \bmod n$
Concatenate each encrypted character to get the encrypted text
```

Fig. 9. Encryption Algorithm in Implemented ECC

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 8, August 2013

In Standard ECC encryption, we consider 'm' having point 'M' on the curve 'E' and randomly select 'k' in the interval of 1 to (n-1). The ciphertext will be generated using the equation $C = m^Q \bmod q$ where 'q' is the prime number in the interval 1 to (n-1). In case of encryption in Implemented ECC, we encrypt each character as $C_i = (A_i)^Y \bmod n$ and concatenate each encrypted character to get the encrpted text. This makes the ciphertext very hard to break. Fig. 9 shows the new encryption algorithm for Implemented ECC.

Algorithm Decryption

```
procedure DECRYPT_MSG
Convert the cipher string into an array of characters.
Encrypt each character $C_i$ as $A_i = (C_i)^X \bmod n$
Concatenate each encrypted character to get the plain text message
```

Fig. 10. Decryption Algorithm in Implemented ECC

In Standard ECC decryption, the original message will be generated using the equation $M = C^d \bmod q$ where 'd' is the private key and 'q' is the prime number in the interval 1 to (n-1). As we saw in key generation, 'd' is a selected prime number and can be easily obtained by permutation. In case of decryption in Implemented ECC, we encrypt each character using $A_i = (C_i)^X \bmod n$ and concatenate each encrypted character to get the plaintext message. Since the private key is generated by using a random number generator algorithm, it will be very difficult to obtain the private key in turn making the job of a attacker harder and the algorithm more secure and efficient.

IV. EXPERIMENTAL RESULTS

We have tested the implementation of Standard ECC and Implemented ECC seperately. The tests are done for the following parametric factors and the results are shown by the following graphs.

1. Cipher Text Size
2. Plain Text Size Versus Cipher Text Size
3. Encryption Time
4. Decryption Time
5. Point Curve Generation

A. *Cipher Text Size in Standard ECC Vs Implemented ECC*

Fig. 11 shows the performance analysis of ciphertext size in Standard ECC Vs Implemented ECC. The Cipher Text Size in Standard ECC shows a normal fluctuation due to varied point curve and point curve arithmetic at the time of encryption. The ideal scenario in ECC must generate a Cipher Text of constant size irrespective of the variation in the point curve arithmetic. Fig 11 gives a numerical proof to this statement.

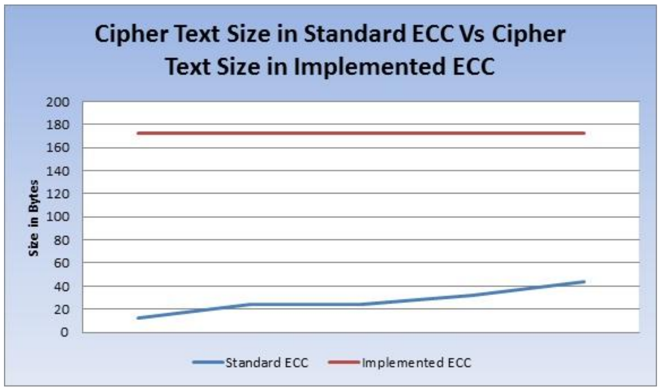

Fig. 11. Cipher Text Size in Standard ECC Vs Implemented ECC

B. *Plain Text Size Vs Cipher Text Size in Implemented ECC*

Fig. 12 shows the performance analysis of plaintext size Vs ciphertext size in Implemented ECC. In any cryptographic algorithm, it is essential to understand the size of the input and the size of output as this is one of the important property of an avalanche effect [11]. Larger the size of the Cipher Text compared with the Plain Text, more secure is the Cipher Text against any Brute-Force attack.

The Implemented ECC upholds this effect and there is no direct relationships between symbols in the Cipher Text to the symbols in the Plain Text (Statistical Analysis). For an input Size of 10 symbols, the output ciphertext generated by Implemented ECC will consist of 25 symbols. So the statistical relationship between the input text and the output is not applicable in case of Implemented ECC.

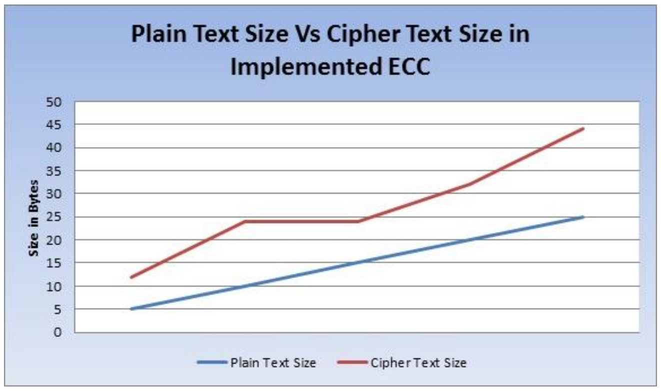

Fig. 12. Plain Text Size Vs Cipher Text Size in Implemented ECC

C. *Encryption Time in Standard ECC Vs Implemented ECC*

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 8, August 2013

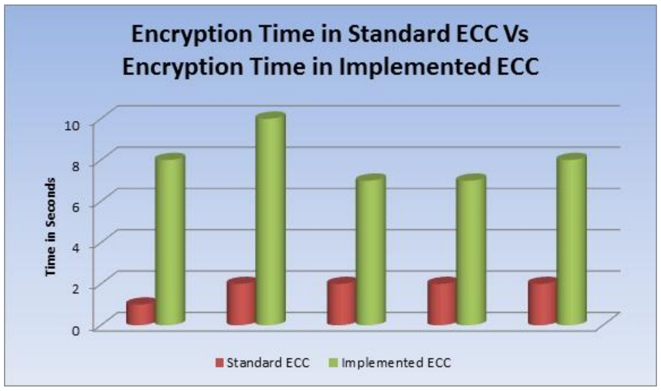

Fig. 13. Encryption Time in Standard ECC Vs Implemented ECC

Fig. 13 shows the performance analysis of encryption time in Standard ECC Vs Implemented ECC. The encryption time has its paramount importance for varied plain text sizes as this determines the time involved in converting plain text into cipher text. Since the Implemented ECC looks into robust method of key generation and random number generation to thrawt the random oracle channel [12], the encryption time is two or three folds greater than the Standard ECC. The Standard ECC as per FIPS does not incorporate any strategy to remove random oracle channel in private key generation. This establishes a subliminal channel from an attacker's perspective to embed his message through a deterministic random number generator. Hence this justifies the security factor encountered in Implemented ECC over Standard ECC and thus encryption for a longer time.

D. Decryption Time in Standard ECC Vs Implemented ECC

Fig. 14 shows the performance analysis of decryption time in Standard ECC Vs Implemented ECC. The decryption time in the Standard ECC and the implemented ECC does not show a larger deviation as the process for decryption is identical in both Standard ECC and the Implemented ECC. This is seen in Section II and Section III of this paper. However, the decryption time in Implemented ECC is slightly greater than the Standard ECC as the key composition is completely different in Standard ECC and the Implemented ECC. The average decryption time in Implemented ECC is 0.5 times or less than when compared with the Standard ECC.

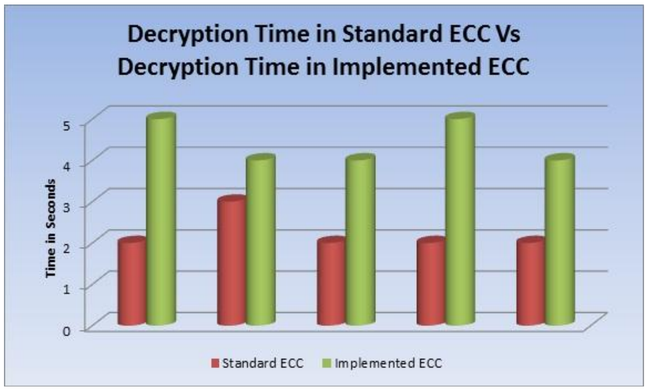

Fig. 14. Decryption Time in Standard ECC Vs Implemented ECC

E. Point Curve Generation in Standard ECC Vs Implemented ECC

Fig. 15 shows the performance analysis of point curve generation time in Standard ECC Vs Implemented ECC. The point curve generation in Implemented ECC is slightly less than the Standard ECC as choosing of points in Implemented ECC is non-deterministic. This factor exhibited by the Implemented ECC shows a visible progress in point curve generation when compared with Standard ECC. The point of using this as a base point in ECC can have different results in different algorithms in a cipher suite like key exchange, signature scheme and the encryption scheme. The point curve generation under Implemented ECC complies to the description mentioned in Niel Koblitz [13] and can be extended further to Jacobian curves [14].

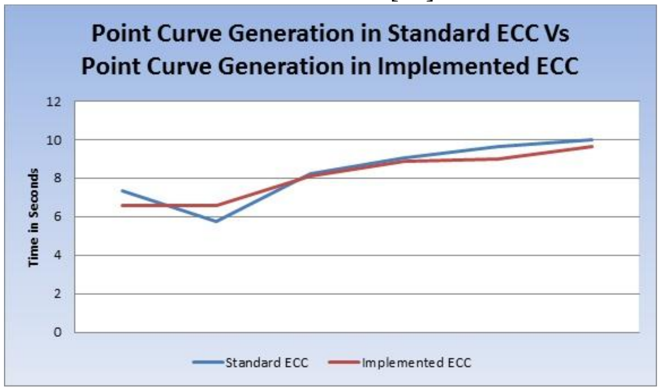

Fig. 15. Point Curve Generation in Standard ECC Vs Implemented ECC

V. Conclusions & Future Work

The work presented in this paper highlights the importance of securing the ECC mechanism over the traditional ECC commercially available in many of the applications. The key pillars of this implementation stands on the key generation and random number generation over the traditional ones. The existing system developed for implemented ECC is used

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 8, August 2013

as an encryption algorithm and can be further extended to key exchange protocol and signature schemes.

ACKNOWLEDGMENT

Gururaja.H.S. thanks M.S.Sudhi, Retired Professor and Head, Dept. of Electronics and Communication Engineering, MSRIT, Bangalore for his guidance and constant support.
Anjan.K. thanks Late Dr.V.K.Ananthashayana, Erstwhile Head, Dept. of Computer Science and Engineering, MSRIT, Bangalore for igniting the passion for research and Late Prof. Harish.G., Erstwhile Assistant Professor, Dept. of Computer Science and Engineering, RVCE, Bangalore for his passion towards research and his motivation to take up research in number theory based algorithms.

REFERENCES

[1] Alfred J. Menezes, Paul C. van Oorschot, and Scott A.Vanstone (1996), Handbook of Applied Cryptography.
[2] Crépeau, C., Slakmon, A. (2003), Simple Backdoors for RSA Key Generation, The Cryptographers Track at the RSA Conference, pp. 403–416.
[3] Shoup, V. (2005), A Computational Introduction to Number Theory and Algebra, Cambridge University Press, Cambridge.
[4] Gururaja, H.S., Seetha, M., Koundinya, A.K. (2012), Covertness analysis of Subliminal channels in legitimate communication, P.S. Thilagam et al. (Eds.): ADCONS 2011, LNCS 7135, pp. 582–591, Springer, Heidelberg.
[5] Gururaja, H.S., Seetha, M., Koundinya, A.K. (2010), A Practical Password based authenication using elliptic curve cryptography, Proceedings of International Conference on Convergence of Science and Engineering, DSI Campus, Bangalore, India.
[6] Young, A. (2004), Malicious Cryptography, 1st edn., pp. 220–240, Wiley Publishing.
[7] Stallings, W. (2006), Cryptography and Network Security, 3rd edn., Pearson Publishing, London.
[8] Chateauneuf, M., Ling, A., Stinson, D.R. (2003), Slope packings and coverings, and generic algorithms for the discrete logarithm problem, J. Comb. Designs 11(1), 36–50.
[9] A Blum, L., Blum, M., Schub, M. (1986), A simple unpredictable pseudo-random number generator, SIAM J. Comput. 15(2), 364–383.
[10]Tezuka, S. (1995), Uniform random numbers: Theory and practice, Kluwer Academic Publishers, Norwell, MA.
[11]H. Feistel, W. A. Notz and J. L. Smith (1975), Some cryptographic techniques for machine-to-machine data communications, Proc. IEEE, 1545–1554.
[12]Kaliski, B.S. (1987), A Pseudo-Random Bit Generator Based on Elliptic Logarithms, Odlyzko, A.M. (ed.) CRYPTO 1986, LNCS, vol. 263, pp. 84–103, Springer, Heidelberg.
[13]Koblitz, N. (1987), Elliptic Curve Cryptosystems, Mathematics of Computation (Vol. 48).
[14]Cantor, D.G. (1987), Computing in the Jacobian of a Hyperelliptic curve, Mathematics of Computation (Vol. 48), 95–101.

BIOGRAPHY

Gururaja.H.S. has completed his Bachelor of Engineering in Computer Science and Master of Technology in Computer Networks from Visvesvaraya Technological University, Belgaum. He is currently pursuing his Ph.D. from JNTU, Hyderabad in the field of Cryptography & Network Security and has around 8 years of teaching experience.

Dr.M.Seetha has completed her Ph.D in Computer Science and Engineering in the area of image processing in December 2007 from Jawaharlal Nehru Technological University, Hyderabad, India. She has a teaching experience of 19 years, presently working as a Professor at GNITS, Hyderabad. She is guiding 10 Ph.D scholars and her research interest includes image processing, neural networks, computer networks and data mining. She has published more than 50 papers in refereed journals and in the proceedings of National/International Conferences and Symposiums. She is the recipient of the AICTE Career Award for Young Teachers (CAYT) in FEB 2009 and received a grant upto 10.5 lakhs over a period of three years by AICTE, India. She is a reviewer for various International Journals and Conferences. She holds the Life Membership of Indian Society for Technical Education (ISTE) and The Institution of Electronics and Telecommunication Engineers (IETE).

Anjan.K.Koundinya has received his B.E degree from Visvesvaraya Technological University, Belgaum, India in 2007 and his M.Tech degree in Department of Computer Science and Engineering, M.S. Ramaiah Institute of Technology, Bangalore, India. He has been awarded Best Performer PG 2010 and Rank holder for his academic excellence. His areas of research includes Network Security and Cryptography, Adhoc Networks and Mobile Computing. He is currently pursuing his Ph.D. from VTU, Belgaum and working in RVCE as an Assistant Professor in the Dept. of CSE.