A NEW VARIANT OF SUBSET-SUM CRYPTOSYSTEM OVER RSA

Sonal Sharma[1], Saroj Hiranwal[2], Prashant Sharma [3]
[1&3]M Tech Student, Sri Balaji College of Engineering & Technology, Jaipur, Rajasthan
[2]Reader (CSE Dept), Sri Balaji College of Engineering & Technology, Jaipur, Rajasthan

ABSTRACT

RSA is an algorithm for public-key cryptography that is based on the presumed difficulty of factoring large integers, the factoring problem. RSA stands for Ron Rivest, Adi Shamir and Leonard, who first publicly described it in 1978. A user of RSA creates and then publishes the product of two large prime numbers, along with an auxiliary value, as their public key. The prime factors must be kept secret. In RSA if one can factor modulus into its prime numbers then the private key is also detected and hence the security of the cryptosystem is broken. The Subset-Sum cryptosystem (Knapsack Cryptosystem) is also an asymmetric cryptographic technique. The Merkle-Hellman system is based on the subset sum problem (a special case of the knapsack problem): An instance of the Subset Sum problem is a pair (S, t), where $S = \{x_1 , x_2 , \ldots, x_n\}$ is a set of positive integers and t (the target) is a positive integer. The decision problem asks for a subset of S whose sum is as large as possible, but not larger than t. This problem is NP-complete. However, if the set of numbers (called the knapsack) is super increasing, that is, each element of the set is greater than the sum of all the numbers before it; the problem is easy and solvable in polynomial time with a simple greedy algorithm. So in this paper we present a new algorithm (Modified Subset-Sum cryptosystem over RSA) which is secure against Mathematical attack, Brute-force attack, Factorization attack and Chosen-cipher-text attack on RSA as well as Shamir attacks. This paper also presents comparison between Modified Subset - Sum Cryptosystem and RSA cryptosystems in respect of security and performance.

KEYWORDS: *Cryptography, Subset Sum, Public key, Private Key, RSA, Merkle-Hellman, Super increasing and Complexity*

I. INTRODUCTION

To solve the problem of secure key management of Symmetric key cryptography, Diffie and Hellman introduced a new approach to cryptography and, in effect, challenged cryptologists to come up with a cryptographic algorithm that met the requirements for public-key systems. Public key cryptography uses a pair of related keys, one for encryption and other for decryption. One key, which is called the private key, is kept secret and other one known as public key is disclosed and this eliminate the need for the sender and the receiver to share secret key. The only requirement is that public keys are associated with the users in a trusted (authenticated) manner through a public key infrastructure (PKI). The public key cryptosystems are the most popular, due to both confidentiality and authentication facilities [1].The message is encrypted with public key and can only be decrypted by using the private key. So, the encrypted message cannot be decrypted by anyone who knows only the public key and thus secure communication is possible.

In a public-key cryptosystem, the private key is always linked mathematically to the public key. Therefore, it is always possible to attack a public-key system by deriving the private key from the public key. The defense against this is to make the problem of deriving the private key from the public key as difficult as possible. Some public-key cryptosystems are designed such that deriving the

private key from the public key requires the attacker to factor a large number. The *Rivest-Shamir-Adleman (RSA)* and *Subset-Sum (Knapsack)* public key cryptosystems [2] are the best known examples of such a system. This paper presents a hybrid cryptography algorithm which is based on the factoring problem as well as Subset-Sum problem called a *Modified Subset-Sum over RSA public key cryptosystem* (SSRPKC).

1.1 Euler's Phi-Function

Euler's phi-function, φ(n), which is sometimes called Euler's Totient Function; find the number of integer that are both smaller than n and relatively prime to n. This function φ(n) calculate the number of element in given set [3].

In this paper, we compare and evaluate the RSA cryptosystem and modified sub-set sum cryptosystem by implementing and running them on a computer. We investigate the issues of complexity, efficiency and reliability by running the algorithm with different sets of values. Moreover, comparisons will be done between these two algorithms given the same data as input.

The rest of paper is organized as follows- section 2, describes the RSA cryptosystem which depends on the factoring large integer numbers. In section 3, describe the introduction of sub-set sum cryptography which depends on the super-increasing order also called NP-complete problem. In section 4, we present our modified algorithm. In section 5, we compare both the algorithm- RSA and modified sub-set sum cryptosystem. A conclusion is shown in section 6.

II. RSA CRYPTOSYSTEM

RSA is based on the principle that some mathematical operations are easier to do in one direction but the inverse is very difficult without some additional information. In case of RSA, the idea is that it is relatively easy to multiply but much more difficult to factor. Multiplication can be computed in polynomial time where as factoring time can grow exponentially proportional to the size of the number. RSA consist of three steps [4]:-

Step 1) Key Generation Process

1. Generate two large random primes, *p* and *q*, of approximately equal size such that their product n = p x q is of the required bit length, e.g. 1024 bits.
2. Compute n = p x q and φ = (p-1) x (q-1).
3. Choose an integer *e*, satisfying $1 < e < \varphi$, such that gcd (e, φ) = 1.
4. Compute the secret exponent *d*, $1 < d < \varphi$, such that $e \times d \equiv 1 \pmod{\varphi}$.
5. The public key is (n, e) and the private key is (n, d). Keep all the values d, p, q and φ secret.
6. n is known as the *modulus.*
7. e is known as the *public exponent* or *encryption exponent* or just the *exponent*.
8. d is known as the *secret exponent* or *decryption exponent*.

Public key (n, e) is published for every one and private key (p, q, d) must be kept secret. Then by using these keys encryption, decryption, digital signing and signature verification are performed.

Step 2) Encryption Process

Sender A does the following: -

1. Obtains the recipient B's public key (n, e).
2. Represents the plaintext message as a positive integer *m*.
3. Computes the cipher text $c = m^e \bmod n$.
4. Sends the cipher text *c* to B.

Step 3) Decryption Process

Recipient B does the following: -

1. Uses private key (n, d) to compute $m = c^d \bmod n$.
2. Extracts the plaintext from the message representative *m*.

2.1 Security of RSA

The security of RSA cryptosystem is also broken by two attacks based on factorization attack and chosen-cipher text attacks [9].

A) Factorization Attack: The security of RSA is based on the idea that the modulus is so large that it is infeasible to factor it in a reasonable time. Bob select p and q and calculate n=p*q. Although n is

public, p and q are secrete. If Eve can factor n and obtain p and q, she can calculate φ = (p-1) x (q-1). Eve then calculate d ≡ 1 (mod φ) because e is public. The private exponent d is the trapdoors that Eve can use to decrypts any encrypted message [9].
B) Chosen-Cipher text Attack: A potential attack on RSA is based on the multiplicative property of RSA [9].

III. SUB-SET SUM CRYPTOGRAPHY

In computer science, the subset sum problem is an important problem in complexity theory and cryptography. The problem is this: given a set of integers, does the sum of some non-empty subset equal exactly zero. For example, given the set {−7, −3, −2, 5, 8}, the answer is yes because the subset {−3, −2, 5} sums to zero. The problem is NP-Complete. There are two problems commonly known as the subset sum problem. The first is the problem of finding what subset of a list of integers has a given sum, which is an integer relation problem where the relation coefficients a_i are 0 or 1. The second subset sum problem is the problem of finding a set of n distinct positive real numbers with as large collection as possible of subsets with the same sum [4].
The subset sum problem is a good introduction to the NP-complete class of problems. There are two reasons for this [4]-

- It is a decision and not an optimization problem
- It has a very simple formal definition and problem statement.

IV. MODIFIED ALGORITHM

In this section we introduce a new approach for public key cryptosystem. Modified Subset Sum (MSS) is an asymmetric-key cryptosystem in which two keys are required: a public key and a private key. Furthermore, unlike RSA, it is one-way, the public key is used only for encryption, and the private key is used only for decryption. Thus it is useless for authentication by cryptographic signing. Modified algorithm consist of three steps-

Step 1) Key Generation Process

1. Generate two large random primes, *p* and *q*, of approximately equal size such that their product m = p x q is of the required bit length, e.g. 1024 bits. (From Big Integer library function of Java)
2. Compute m = p x q and φ = (p-1) x (q-1).
3. Choose an integer *e*, satisfying 1 < e < φ, such that gcd (e, φ) = 1.
4. Compute the secret exponent *d*, 1 < d < φ, such that e x d ≡ 1 (mod φ).
5. Choose a superincreasing set A = (a_1, ..., a_n)
6. Choose an integer M with M > $SUM_{i=1...n}(a_i)$. M is called the modulus.
7. Choose a multiplier W such that gcd(M, W) = 1 and 1 <= W < M This choice of W guarantees an inverse element U: UW = 1 (mod M)
8. To get the components b_i of the public key B, perform $b_i = a_i$*W mod M, I = 1 ... n

The superincreasing property of A is concealed by *modular multiplication*.
The public key is (B, n, e) and the private key is (A, M, W, n, d). Keep all the values d, p, q and φ secret.
Public key is published for every one and private key must be kept secret. Then by using these keys encryption and decryption are performed.

Step 2) Encryption of Message

Sender A does the following: -

1. The length of a message to be encrypted is fixed by the parameter n prior to encryption; a possibly larger message p has to be divided into n-bit groups.
2. Let p = (p_1, p_2... p_n) the message to be encrypted.
    - The ciphertext c is obtained by computing $c = b_1p_1 + b_2p_2 + ... + b_np_n$
    - Computes the cipher text $c_1 = c^e$ mod n.
    - Sends the cipher text c_1 to B.

Step 3) Decryption of Message

Recipient B does the following: -

1. Uses private key and first compute $m_1 = c_1^d \bmod n$.
2. First compute $c' = Um_1 \bmod M = W^{-1}c \bmod M$
3. Now solve (A, c'). Because A is superincreasing, (A, c') is easily solvable. Let $X = (x_1 ... x_n)$ be the resulting vector and $p_i = x_i$ and $p = (p_1 ... p_n)$ is the plaintext.

V. COMPARISON OF RSA AND SSRPKC CRYPTOSYSTEMS

5.1 Simulation Results

For the simulation purpose, SSRPKC cryptosystem is implemented as a user-friendly GUI. This GUI application is implemented using Java Big Integer library functions [5]. In this application, one can either enter the prime numbers or can specify the bit length of the prime numbers to generate automatically. Big Integer library provides operations for modular arithmetic, GCD calculation, primarily testing, prime generation, bit manipulation, and a few other miscellaneous operations. The simulation of the algorithm, implemented in JAVA [5], running on a 2 GHz P-IV Processor and 512 MB RAM, using a 1000 characters long message for encryption/decryption. The PKC algorithms (RSA & SSRPKC) have some important parameters affecting its level of security and speed [6]. The complexity of decomposing the modules into its factors is increased by increasing the module length. This also increases the length of private key and hence difficulty to detect the key. Another parameter is the number of items in set A. As the number of items in set A increases, the size of the message which is encrypted at a time also increases, hence the security also increases as well as difficulty of detecting the private set A from public set B also increases. The RSA and SSRPKC parameters are changed one parameter at a time and the others are kept fixed to study the relative importance. The key generation, encryption, decryption time is depends on the speed of the processor and the RAM. Table 5.1 shows the simulation results of both the algorithms.

5.1.1 Changing the modulus length:

Changing the modulus affects the other parameters of the algorithms as shown in Table 5.1. It is clear here that increasing the modulus length (bits) increases the bit length of their factors and so the difficulty of factoring them into their prime factors. Moreover, the length of the secret key (d) increases at the same rate as n-bit increases. As a result, increasing the n-bit length provides more security. On other hand by increasing the n-bit length increases the values of key generation time, encryption time and decryption time. Hence increasing the n- bit length increases the security but decreases the speed of encryption, decryption and key generation process as illustrated by Figure 5.1 and 5.2.

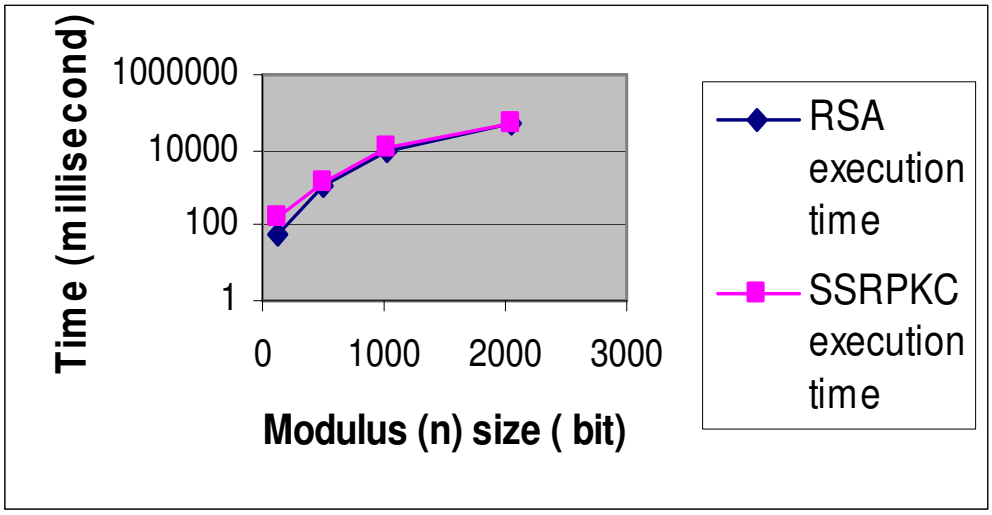

Figure1. Modulus size v/s RSA & SSRPKC algorithm's execution time, taking number of items in set A 128 and size of Public Key 128 bit.

Table 1: Effect of changing the modulus length and size of set A on the size of Private key, Key generation time, Encryption time and Decryption time, while the size of Public key has kept constant (128 bit)

Size of N (bit)	Size of d (bit)	Number of element in set A	SSRPKC				RSA			
			key generation time (ms)	Encryption time (ms)	Decryption time (ms)	Total Execution time(ms)	Key Generation time (ms)	Encryption time (ms)	Decryption time (ms)	Total Execution time(ms)
128	128	32	16	235	78	329	16	94	62	172
128	128	64	15	94	47	156	0	16	47	63
512	512	32	125	1203	1766	3094	109	563	1719	2391
512	512	64	63	344	875	1282	63	141	859	1063
512	512	128	78	172	453	703	47	78	422	547
1024	1024	32	688	5407	11328	17423	688	1719	12172	14579
1024	1024	64	453	6593	5735	12781	453	2968	5688	9109
1024	1024	128	562	516	2859	3937	515	219	3344	4078
1024	1024	512	12812	187	781	13780	281	47	735	1063
2048	2048	32	3735	9563	85140	98438	3719	3688	85672	93079
2048	2048	64	1563	3625	42437	47625	1563	1688	43234	46485
2048	2048	128	7125	6266	20734	34125	7078	3829	21406	32313
2048	2048	512	17797	797	5281	23875	7703	375	6172	14250
2048	2048	1024	29704	422	2797	32923	2891	203	3406	6500

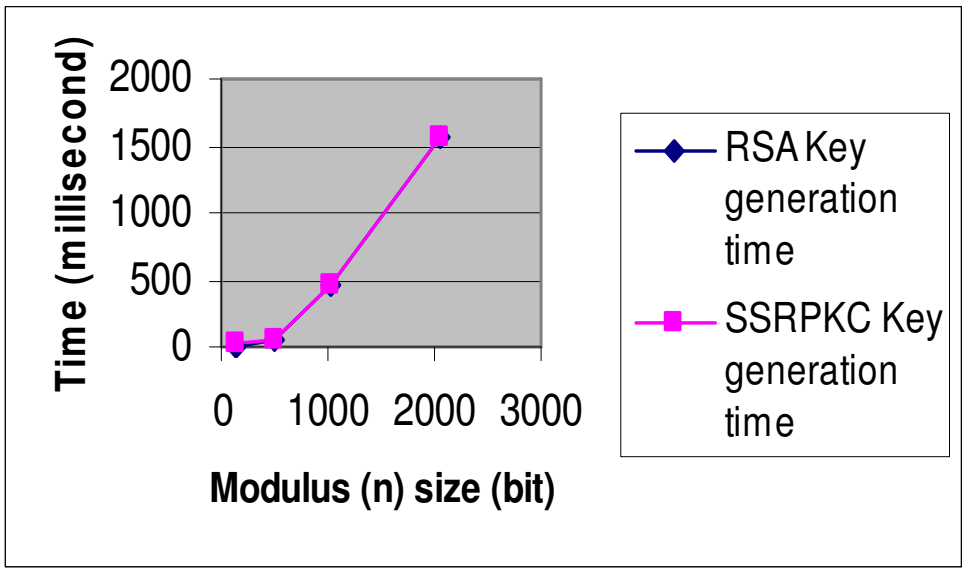

Figure 2. Modulus size v/s Key generation time, taking size of Public key 128 bit and number of items in set A are 128

5.1.2 Changing the number of items in set A:

On the basis of simulation results of Table 5.1, following Figure 5.3 shows the effect of number of items on encryption and decryption time of both the algorithms. Here key generation time of SSRPKC algorithm depends on the number of items in set A and as the number of items increases key generation time also increases that's why for more then 500 items in set A, execution time of SSRPKC algorithm also increases. RSA algorithm's execution time doesn't depend on set-A items.

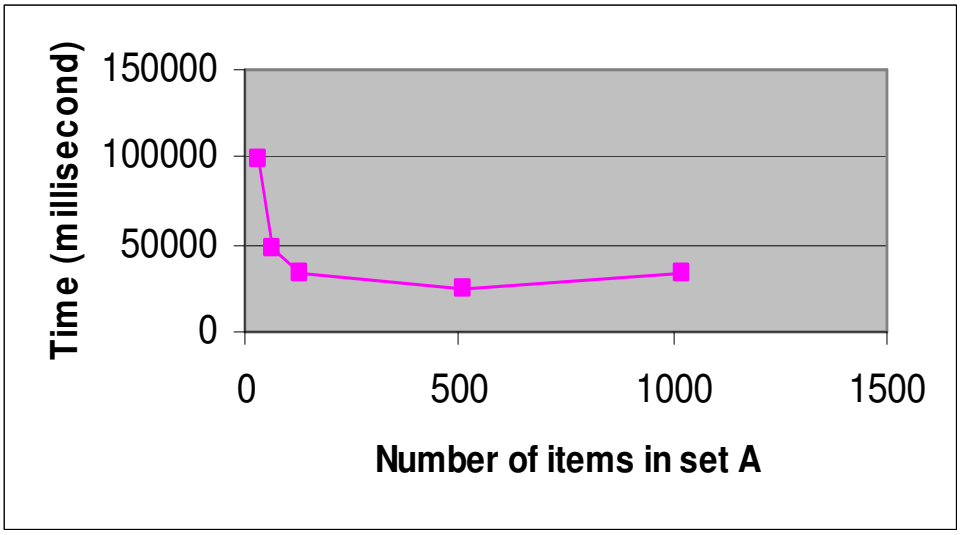

Figure 3. Number of items in set A v/s SSRPKC algorithm's execution time, taking size of modulus 2048 bit and size of private 128 bit.

5.2 Complexity analysis of SSRPKC cryptosystem

The RSA cryptosystem is well-known asymmetric cryptosystem in which the computational complexity for both of the encryption and the decryption are of the order of k^3, where k is the number of bits of the modulus n [7]. The computational complexity of subset sum problem is *O(k),* if a super-increasing set with greedy approach is used where k is the number of items in set-A [8]. So in this way, computational complexity of SSRPKC cryptosystem is on the order of $O(k + k^3)$ i.e. $O(k^3)$ or encryption and on the order of k^3 for decryption. So the computational complexity of SSRPKC is equivalent to RSA cryptosystem. The simulation results of both the algorithms shows that the execution time of SSRPKC is and 1.2 times more than RSA.

5.3 Security analysis of SSRPKC cryptosystem

Two possible approaches to attacking the SSRPKC / RSA algorithms are as follows [1, 9]:

1. *Brute force:* This involves trying all possible private keys.
2. *Mathematical attacks:* These are based on factoring the product of large primes such as factor n into its prime factors p and q respectively then calculating φ, which, in turn, enables determination of $d = e^{-1} \bmod \varphi$.

Estimated resources needed to factor a number within one year are as follows [9].

Table 2: Recourses needed to factor a number within one year

Size of number (Bits)	PCs	Memory
430	1	128 MB
760	215,000	4 GB
1020	342×10^6	170 GB
1620	1.6×10^{15}	120 TB

As the SSRPKC cryptosystem is based on the hybrid combination of the subset sum problem and RSA cryptosystem, so one have to factor the modulus into its primes as well as find the secret set-A to break the SSRPKC algorithm [2, 7]. If RSA which is based on single modulus, is broken in time x and Subset sum based algorithms is broken in time y then the time required to break SSRPKC algorithm is $x*y$. So the security of SSRPKC algorithm is increased as compare to RSA algorithm and it shows that the SSRPKC algorithm is more secure for *Mathematical attacks* [1].

As in SSRPKC double decryption is performed and unlike RSA that is not only based on private key but also based on the subset sum problem so one can't break SSRPKC only guessing the private key only. So it shows that SSRPKC algorithm is more secure as compare to RSA for *Brute force attack* [1].The RSA cryptosystem is based on the Integer Factorization which is break by the Pollard rho method [4], Pollard p-1 method [4] and Fermat method [4]. Because of these methods the security of RSA cryptosystem is broken by the Factorization attack [3] and Chosen Cipher Text attack [3].

As the SSRPKC cryptosystem uses dual modulus, so for breaking this one have to factor both the modulus. That's why our cryptosystem provides the far better security against the factorization methods and attacks on RSA cryptosystem.

VI. CONCLUSION

This paper presents an extended version of Subset-Sum problem over RSA algorithm called Extension of cryptosystem through subset-sum over RSA. It relies on facts of mathematics that indicates that given a very large number it is quite impossible in today's aspect to find two prime numbers whose product is the given number. The size of number increases, the possibility for factoring the number decreases. In RSA, if one can factor modulus into its prime numbers then he can get the private key too. To improve the security, SSRPKC cryptosystem is developed. The disadvantage of new cryptosystem is that, unlike RSA it can not be used for authentication as it is based on the one way function. Another disadvantage is the slow down of execution process as compare to RSA. But it is clear from the simulation results that it is more secure than the RSA algorithm and our cryptosystem provides the far better security against the factorization methods and attacks on RSA cryptosystem.

REFERENCES

[1] William Stallings,"Cryptography and Network Security", ISBN 81-7758-011-6, Pearson Education, Third Edition, pages 42-62,121-144,253-297.
[2] Ralph C. Merkle, Martin E. Hellman. "Hiding Information and Signatures in Trapdoor Knapsacks", IEEE Transactions on Information Theory, vol. IT-24, 1978, pp. 525-530.
[3] Behrouz A. Forouzan, Debdeep Mukhopadhyay, "Cryptography and Network Security", 2nd Edition, TMH.
[4] Alfred J Menezes, Paul C van Oorschot, Scot A Vanstone, Handbook of Applied Cryptography, CRC Press, 1997.
[5] Neal R. Wagner, "The Laws of Cryptography with Java Code", Technical Report, 2003, pages 78-112.
[6] Allam Mousa , "Sensitivity of Changing the RSA Parameters on the Complexity and Performance of the Algorithm", ISSN 1607 – 8926, Journal of Applied Science, Asian Network for Scientific Information, pages 60-63,2005.
[7] RSA Laboratory (2009), "RSA algorithm time complexity", Retrieved from http://www.rsa.com/rsalabs/node.asp?id=2215 (4 April 2011).
[8] Adi Shamir, A Polynomial Time Algorithm for Breaking the Basic Merkle-Hellman Cryptosystem. CRYPTO 1982, pp279–288.
[9] CHUK, Chinese university (2009), "RSA Algorithm security and Complexity", Retrieved from http://www.cse.cuhk.edu.hk/~phwl/mt/public/archives/old/ceg50 10/rsa.pdf (26 Jan. 2011)
[10] Wenbo Mano, "Modern Cryptography Theory and Practice," Prentice Hall.
[11] Bryan Poe, "Factoring the RSA Algorithm", Mat / CSC 494, April 27, 2005, pages 1-6.
[12]Adi Shamir, A Polynomial Time Algorithm for Breaking the Basic Merkle-Hellman Cryptosystem. CRYPTO 1982, pp279–288
[13] A Method for Obtaining Digital Signatures and Public-Key Crypto Systems by R.L. Rivest, A. Shamir, and L. Adleman.
[14] Menezes, A. J., Van Oorshot, and Vanstone, P.C. S.A. Handbook of Applied Cryptography, CRC press, 1997.
[15] B. Schneizer, Applied Cryptography, New York: John Wiley, 1994.

[16] http://www.certicom.com, "The Elliptic Curve Cryptosystem," September 1997, (dated 02-04-2010)
[17] J H Moore, Protocol failures in Cryptosystems, Contemporary Cryptology, The science of Information Integrity, Ed. G J Simmons, 541-558, IEEE Press 1992.
[18] Diffie, M Hellman, "New Directions in Cryptography", IEEE Transactions on Information Theory, Vol 22, 1976.

Author's Biography

Sonal Sharma was born on 23 May 1984. She is M.Tech. Student in SBCET, Jaipur (Rajasthan). She has completed B.E. (I.T.) in 2006 from University of Rajasthan, Jaipur.

Saroj Hiranwal was born on 20 Jan 1982. She has done B.E.(I.T.) in 2004 and MTECH(CSE) in 2006. Her teaching Experience is 6.5 year in the organization - Sri Balaji College of Engineering and Technology, Jaipur with the designation of Reader and HEAD.

Prashant Sharma was born on 04 June 1985. He is the M.Tech. Student in SBCET, Jaipur (Rajasthan). He has completed B.E.(I.T.) in 2007 from University of Rajasthan, Jaipur.