www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume1 Issue 3 Dec 2012 Page No. 125-136

Symmetric Key Management: A new approach

***Santosh Deshpande*[1]**

B V Bhoomraddi of Engineering and Technology, Hubli 580 031, Karnataka, India
Corresponding Author (e-mail : sldeshpande@gmail.com)

ABSTRACT

Many cryptographic algorithms (e.g., AES, HMAC) require the establishment of shared keying material in advance. The Federal key agreement schemes document is based on ANSI X9.42 agreement of symmetric keys using discrete logarithm cryptography, ANSI X9.44 key agreement and key transport using factoring-based cryptography and ANSI X9.63 key agreement and key transport using elliptic curve cryptography. It is necessary for the keying material to establish and maintain cryptographic keying relationships. Shared secret is a value computed using a prescribed algorithm and combination of keys belonging to the participants in the key establishment scheme. The public key algorithms like RSA are becoming the standards for the encryption. But the major threat of using RSA is, it is multiplicative and hence does not guarantees the uniqueness of the key and further it does not stand true to the dynamic demands of the system. Practically it is proved that such public key algorithms are slow as compared to the symmetric key algorithms. The PKI has provided a strong platform for the RSA algorithm. In view to overcome above limitations in this article a cryptographic method of key management has been proposed that uses the symmetric key algorithm (Blow fish) over the asymmetric key algorithm (RSA algorithms). The symmetric key algorithms do not have any key management scheme. Further, the proposed work uses ant colony based evolutionary computing algorithm to meet the requirement in which the shared secret value changes dynamically with time and matures with the time as numbers of communications amongst the legitimate users increases. The method proposed has been found have system of key management, which apart from being faster has all the security features. The efficiency in time management in all the aspects namely Key Generation, key Transmission, Key Revocation, Encryption and decryption is found to be much better than the widely used PKI. This speed do not compromise the security aspects of the system, in fact this efficient time management has helped to build up the secured transmission.

Practically to avoid the time of encryption and decryption using the public key infrastructure certifying authorities use the session keys that use symmetric key algorithm. This method is not providing any authentication mechanism. The proposed system stands balanced and solves all the problems mentioned above. Overall it is a key management scheme for the symmetric key algorithms.

KeyWords: Public Key infrastructure, Symmetric Key algorithm, Key Management, RSA, Evolutionary computing

1.0 Introduction:

Private networks are expensive and face the problem of physically securing their lines. Virtual Private Networks (VPN) offer an economical solution to this problem by establishing a virtually private network while physically sharing the Internet media. Privacy is achieved by creating a cryptographically secure tunnel between authenticated peers, making the traffic virtually invisible to the rest of the Internet but leads to the issue of key management.

The protocols like Diffie-Hellman, Oakley key exchange, SKEME and IKE are not presently the universal standards. [1] The Deffie Hillman protocol fails to authenticate the legitimate users. The man in middle is also one such threat to this method [6,7,8]. IKE does not provide any management of certificates or long-term keys. Thus, in general, the shared secret should be exchanged offline. [9]

Also the public key algorithms like RSA are becoming the standards for the encryption. But the major threat of using RSA is, it is multiplicative and hence does not guarantees the uniqueness of the key [2] and further it does not stand true to the dynamic demands of the system. [10]

The widely used existing key management infrastructure is PKI. In this, the explanation about the key management does not specify about how the key (private key) will be transmitted. The only possible explanation is that, this secret key is transmitted through secure socket layer. The PKI has several such pitfalls [3,10].

In view of the above a new approach seemed warranted which can over come the problems mentioned above and in this paper such a system is presented in which the key management scheme uses the symmetric key algorithms. The issue of key management is handled using three separate modules. In the first, the connection establishment with the trusted party is achieved in which there is no involvement of the certifying authority like PKI [11]. In the second module the sharing of keying material amongst the authorized parties is established through sharing of graph, which helps for the generation of the key as well as the acknowledgement. This module will take care of the authentication as well as key generation within the legitimate users. The keys shall be obtained using meta-heuristics of Ant Colony Algorithm [4] that matures with time and produces accurate results. The evolutionary computing pusses a strong property of generating accurate results. The algorithms use heuristics that produce no colliding of the values, which is the basic property of an ideal key generator. Further, this property is used to generate keys dynamically in the propped method as against PKI. The user needs to remember only the source and destination of the nodes that is two-digit number each, in the proposed method while public keys are large and thus are difficult for people to read, write, speak, memorize, or compare. Further, public keys [12] which are transient, are for single-use and need to be revoked when the private key is lost or stolen, and should be changed periodically, while it is good practice to use different key pairs for different protocols or in different devices, so that a user will have multiple public keys as is being possible in the proposed system. In the third module the new set of shared secret is generated periodically.

The proposed method uses ACO based distributed problem-solving technique for the key generation as well as for authentication. The encryption of the plane text is done with the help of secure symmetric key algorithms like Blow Fish (a 64-bit block cipher with variable length key (0-448bits) and 16 rounds), which encrypts at a very high rate (~4 times faster than DES) [5]. In the presented paper key management is a set of techniques and procedures supporting the establishment and maintenance of keying relationships between authorized parties.

2.0 The aspects of System Design

The methodology of achieving keying material will be efficient using graphs because of their transitively closed nature. This has been used in the present work to produce efficient keying system. The Signing vertices and edges of a dynamically growing, transitively closed graph motivates transitive signature which was used first by Micali and Rivest [14]. The general designing paradigm proposed involved an underlying standard signature scheme, which is required to be existentially non forgeable against adaptive chosen message attacks. The requirement for the underlying signature is not necessarily so strong; instead non-adaptive security is enough to guarantee the transitive signature scheme secure in the strongest sense, i.e., transitively are non forgeable under adaptive chosen message attack. This paper uses such transitive signature schemes, and also proposes a specific transitive signature scheme based on factoring. Hence the choice of standard signatures that can be employed by transitive signature schemes is enlarged. The efficiency of transitive signature schemes may be improved since efficiency and security are trade-off parameters for standard signature schemes.

A typical transitively closed graphs G is presented in figure 1 in which $(u,v) \epsilon E$ and $(v,w) \epsilon E \rightarrow (u,w) \epsilon E$. The transitive signature scheme in the figure is achived by signing vertices ($\sigma (v)$) and edges ($\sigma (u,v)$) such that given $\sigma (u,v)$ and $\sigma (v,w)$ can compute $\sigma (u,w)$.

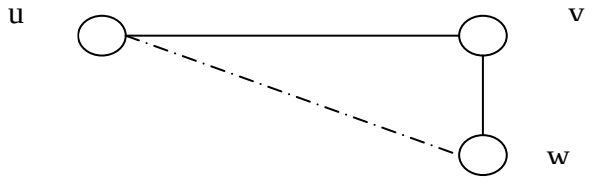


Figure 1 ransitively closed graph

Imagine some issuer signs various vertices and edges over time as shown in figure 1. Inferred signature σ (u,w) should be indistinguishable from an oreginal issuer's signature. This provides the efficient issuing and verification of the signature.

In short Graph $G_1 = (V, E_1)$ is a transitive closure of the graph G = (V, E) iff $\forall i,j \in V$ (there is a path between *i* and *j* in *G* ➔ $(i,j) \in E_1$) (figure 2)

Example:

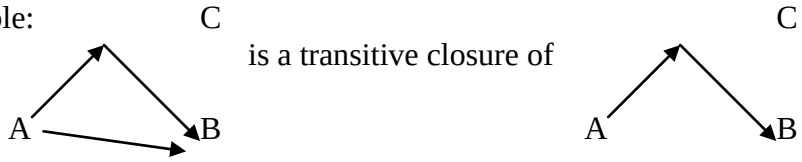


Figure 2 Example of Transitively closed graph

Security aspects of transitively closed graphs being

((x3-x2),(y3-y2))= ((x3-x1),(y3-y1)) + ((x2-x1),(y2-y1))σ(2,3)= σ (2,1)+ σ (1,3).

Which are represented in figure 3.

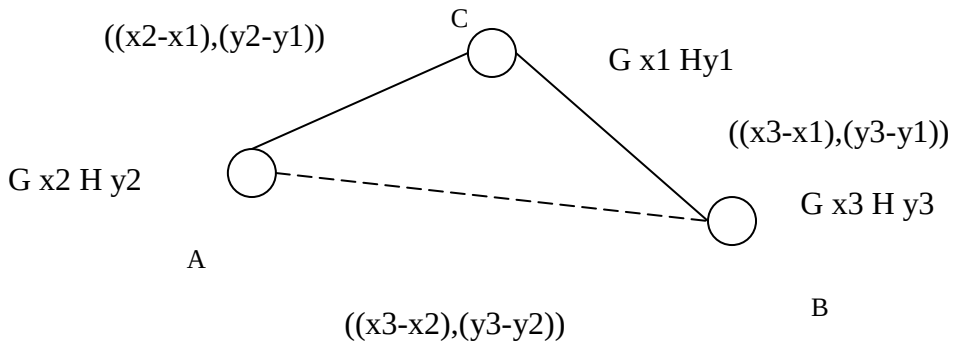


Figure 3 Transitively Closed Graph

It is very difficult to find new function σ (2, 3) that combines σ (2, 1) and σ (1, 3) that is combination of the two or more functions that give the same results and this probability is very less. This cannot be figured out because the numbers of functions and the edge values are unpredictable.

In general let V_1, V_2. V_3 .V_4 ….V_n be the vertices and e_1, e_2 ….. e_m belong to their edge set. The graph will be like a seed of keys that will be shared by the legitimate users. The Figure 4 explains the graph in detail with n = 5 and m = 10. Assuming nodes have functions$f_1( )$, $f_2( )$, $f_3( )$, $f_4( )$ and $f_5( )$ respectively attached to the vertices V_1, V_2. V_3 .V_4 .V_5 with e_1, e_2 ….. e_{10} being the weighed edge set of the sample graph. As the edge sets are traversed by the artificial ants the value is passed from one node function to other node function. The key value changes as the nodes are traversed, for example, if ant traverses V_1 to V_5 the value will be $f_5(f_1(e_1))$ and so on.

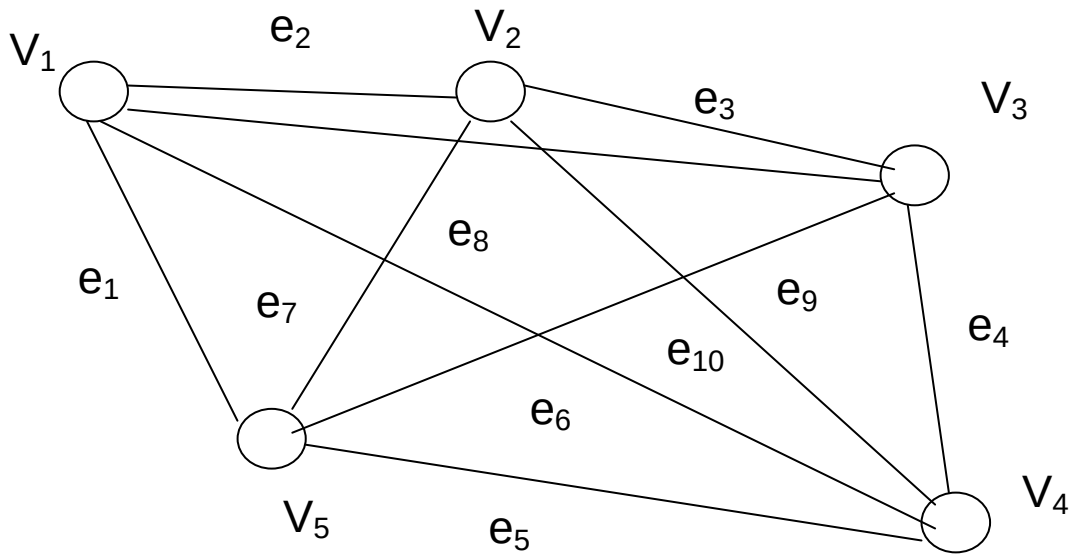
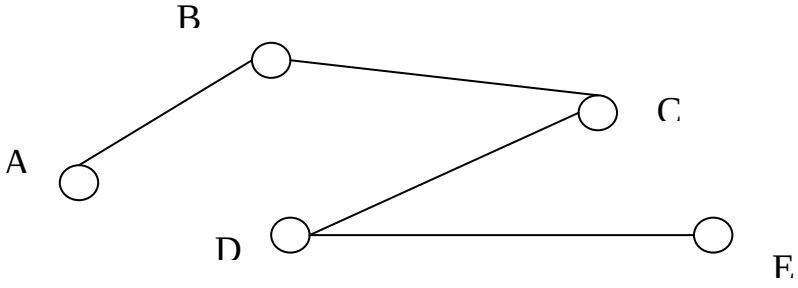


Figure 4 The Proposed Graph

All the pubic key generation algorithms deal with the problems like factorization and discrete mathematics of higher values of the prime numbers. These methods are highly time consuming and vulnerable to the attacks. Thus, the mathamatical functions used above are other than discrete and factorization type. This is being to avoid longer computational time and possibe intusion as is being done in earlier methods discussed above. The graph so designed will solve the problem as it can use any function that is sensitive to the input values like logarithmic and hyperbolic functions. These functions are invoked when that node is traversed while generating the key value. The ants to solve the TSP will use such graphs. With this the functions at the nodes will generate the keys and will be stored in the *tabu_list* of the ants. After the global optimization the final list of key values will be available in the *tabu_list*. The above discussion is represented through the figure 5

In the figure 5, ABCDE are the nodes of one tour. The designed graph says that every node is a function and edge has some weight generated randomly. The key will be generated as follows. If the user chooses the source node as B and destination node as node E then key will be



$$\text{Key} = f_e(f_d(f_c(f_b(BC),CD),DE)) \quad \ldots\ldots\ldots 1$$

Figure 5 The chosen Path

In the above formula the suffix of f's are the functions associated with individual nodes. These functions will collectively generate the value of a key, which is unique and highly sensitive to its seed value as the chosen functions are logarithmic and hyperbolic. Further, the key generation is expected to be much faster than the RSA key generation in view of the inherent different search

methodologies involved. In fact this point is taken in the design aspect has been shown to be the case, which is presented under the section results and discussion. Naturally, key values are to be pseudo random in nature and have to pass the randomness criteria set by NIST. The same concept can be used for acknowledgment by applying the functions to the values of dynamically changing pheromone levels. The proposed design expects the client as well as host need to be synchronized. The explanation about the system synchronization is explained herewith.

Let us assume two systems system 1 and system 2 between which the communication has to be established. At the time of key generation in system 2 there will be a difference in pheromone level between both the systems (the sending system and the receiving system) and hence the system synchronization becomes essential. . In order to account for this, pheromone level is decremented by the time taken to generate all the tours and the network delays, both of which were calculated in the earlier steps. The proposed synchronization protocol is presented in figure 6, which is self-expiatory.

In the following paragraphs the other modules of system and key managements are discussed.

Thediscussion is focused with special attention on issues of the key management like Distributing keys, establishing a shared key with another party, Key storage, Revocation, so as to meet the criterions suggested by the Internet Key Exchange policy document.

Figure 7 give the overall design proposed, it's features and the working of the proposed model. The model strongly proposes the concept of the use of same set of keys shared by the legitimate users. That means the same information is available with the system, which want to communicate with each other. The data is the graph that the systems will share.The graphs are generated by the individual system assigning random weights and functions and the existing PKI is used to transmit this graph. Only sharing of the graphs doesn't serve the purpose. At the individual machine levels the graphs are generated using the seed values. The seed values are highly system specific like time. This defines the uniqueness of the seed values. This seed value will generate the undirected graphs. The user must specify the numbers of node values.

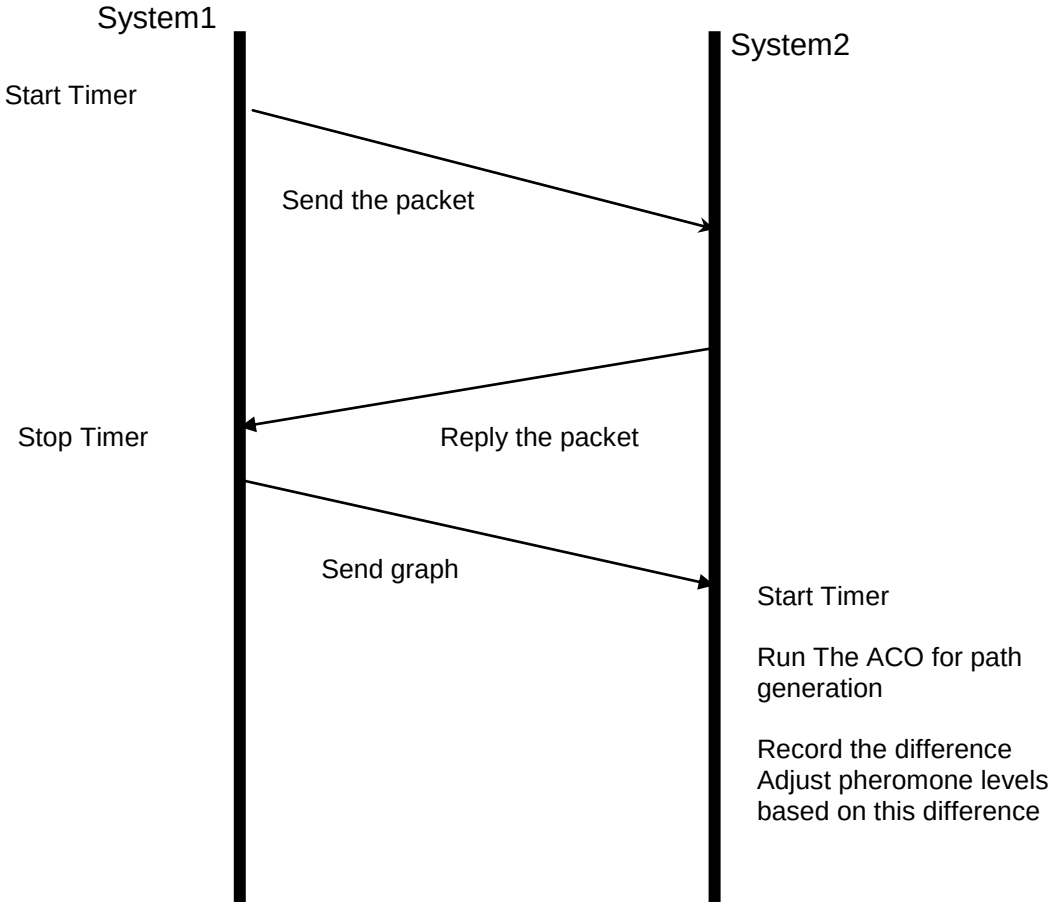


Figure 6 Time Synchronization

These node values will be also one of the parameters of the seed value of the graph. Now once generated, the graph is ready to solve the Traveling Salesman Problem with a relation between the number of nodes and number of tours being, number of nodes equals to Hamiltonian cycles in the Graph. This means the system runs the Ant Colony optimization for the graph at its worst-case complexity as given below.

$$\text{ACO complexity} = tn^3$$

Where t = Number of Tours and n = Number of nodes

In the worst case, when t = n to overall complexity of the tours will be n^4

If the number of nodes input by the user will be a large value then the time required for the ACO to get all the tours will increase heavily in the order of power of four of the numbers of nodes. Hence it's better that the user sticks on the lower values. This may raise one question about the time for the key generation of RSA algorithm. But here the numbers of keys generated will be $n^2 - n$ that are sufficient in numbers.

The Ant colony will be modified once all the tours are received. These will be stored separately in the *tabu_list* of the ACO. The pheromone level list is also a new modification added in the ACO. The *pher_list* is another data structure that is storing the edge and pheromone linkage. The evaporation factor ρ will be also decided by the user. Once all the tours are available then a packet will be formed which shall contain the graph, the current pheromone levels, the decay factor ρ and the timer.

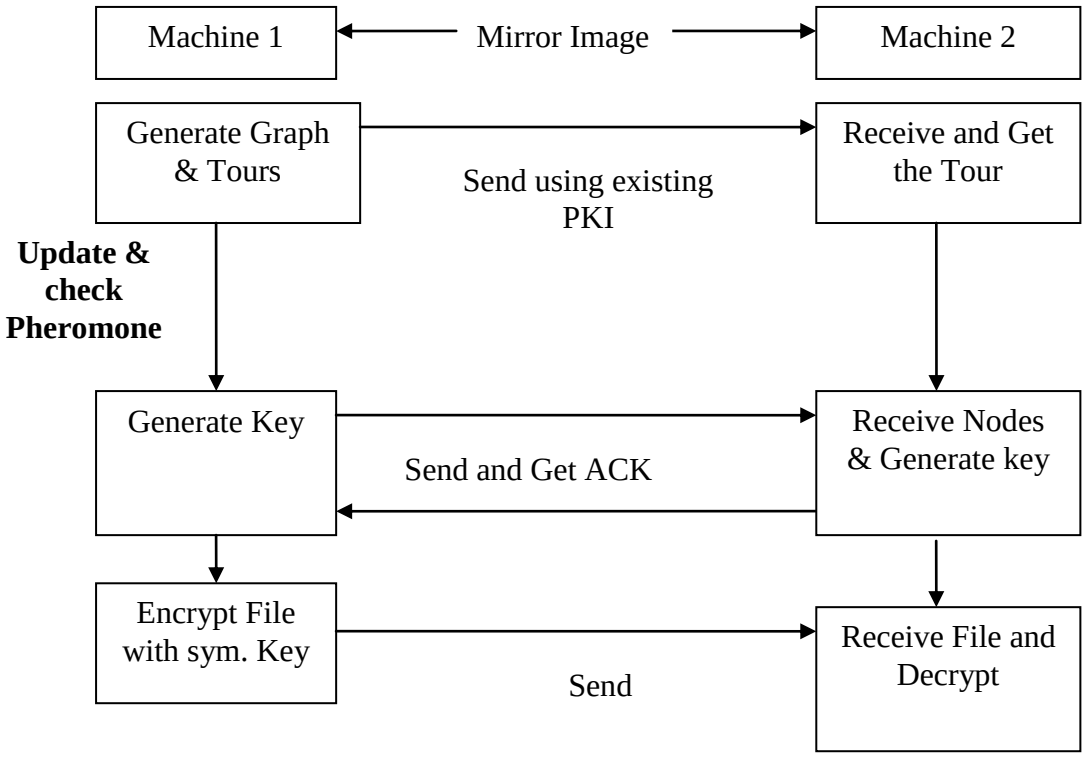


Figure 7

he timer is used for the synchronization of the legitimate users. The algorithm used is explained above and the sender receives the network traffic at that instance. Then the timer will be used for the updating the pheromone levels. Every one minute the pheromone level is updated with the decay factor. This pheromone levels on the edges will act as the acknowledgement. This packet will be sent from one machine to the other by encrypting the packet with RSA algorithm. This packet size will be less than 2 KB. The encryption using RSA will be fast as the file size is less than 2KB. This packet will be sent to the receiver and he will decrypt the packet to get all the details.

Then he will run the ACO to gain the key set. Then the acknowledgement will be sent.

Now the set up is ready and timers are running independently. Pheromone levels are getting updated simultaneously producing complete mirror image of the systems.

If the user wants to communicate then he will select two node values. These values will be searched in the *tabu_list* and the longest path is identified. Then the edges traversed and functions in the node will generate the key values. But the key values are not transmitted in the network. On the other hand the initial node and final node is sent without encrypting. The receiver will acknowledge the transmitter based on the pheromone levels on that path. At this time different functions are applied by the nodes on the pheromone level values and not the ones, which are used for the key generation. This acknowledgement is sent back and it is verified at the sender end. Once the nodes are acknowledged, the file encrypted with the blowfish will be sent to the receiver. While receiving the file it will be decrypted and stored in the location desired by the user. Acknowledgements are received at every transaction. The new graph will be shared well before the pheromone level reaches zero value. At this point the new graph will be shared by using the previous graph only but not PKI i.e PKI is used only at the initial stage.

Key management life cycle starts with key generation and the designed graph is capable of generating these keys. As stated earlier distribution process is done by the PKI only once and further care is taken that direct key values are never sent on the network. The key installation will happen as soon as the ACO is run by the systems and both are acknowledged. The key is used there after till the pheromone level reaches zero. This ensures that the life span of the keys is limited after which the set of keys will be terminated. Overall it covers all the aspects of the key management. The system can be attacked only with brute-force method as it the length of the key is assured to be more than 10 digits leading to a requirement of longer time.

One of the important points to be noted is the controlling mechanism of the keying material in the proposed work. No mechanism is available to change the key values in the existing schemes, but in the proposed scheme presented in this paper, at any given time if the user feels like changing the set of key values it can be achieved without much difficulty. At the same time keys remain valid for the finite set of time and it becomes difficult for the intruder to attack. At the same time the dynamic acknowledgements will also ensure that the secure communication takes place between the legitimate users.

Apart from the above one finds the advantage of the proposed methodology over the earlier ones as below.

- In the existing scheme the updating and revocation and destruction involves communication overhead but in the proposed system such communication overheads do not exist.

• There is no necessity of storage of the key value as the file gets decrypted as it is getting received.

3.0 Modular Testing and Results:

The proposed scheme was exhaustively tested at each phase with respect to security as well as time of the key generation. The results are presented and discussed below.

3.1 Key generation

The first parameter is the generation time of the keys. The time for RSA key pair will be obviously more because the RSA has to undergo several exponential and factorizing and modular arithmetic functions. The results of this work i.e of the key generation are presented in table-1. A look at table-1 shows that the average time for key generation of RSA algorithm is much higher than the ACO algorithm. In the present work the average time for key generation of RSA algorithm is found to be 5648.6 μSec while for ACO algorithm proposed in the present method is only 330.45 μSec, in spite in the present method n^2 –n keys are produced as against one pair in RSA. Thus, the average speed enhancement in terms of the average time is 17.09366. The graphical representation (figure 7) clearly indicates the above observation. It may be observed from this figure that the randomness of the time is much higher in RSA as compared with the current ACO approach.

Table -1

Iterations	RSA algorithm (μSec)	ACO Algorithm (μSec)
1	13962	443
2	535	393
3	12347	341
4	7745	246
5	5048	346
6	2291	297
7	2740	343
8	526	236
9	6339	246
10	3274	378
11	1470	389
12	5191	341
13	1764	257
14	8284	310
15	8150	422
16	7775	378
17	3653	272
18	3621	297
19	9907	327
20	8350	347

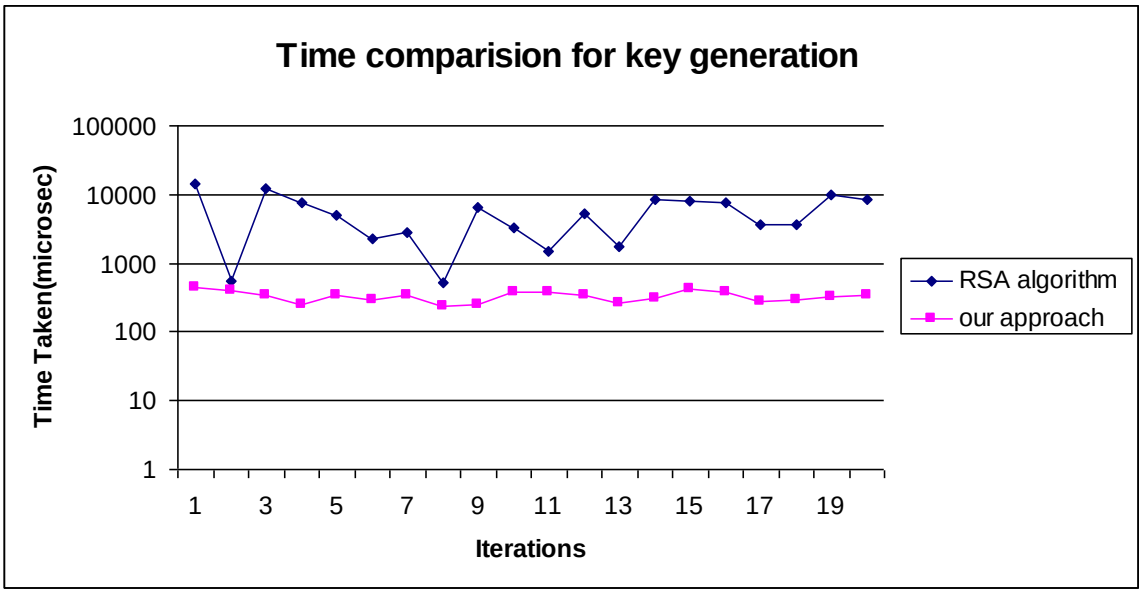


Figure 7 Time comparison of key generation

The through comparison of RSA with Blowfish was also done as a modular approach. This is very necessary as PKI is considered to be a standard and it uses the RSA algorithm, while the present method uses symmetric key approach, which is much faster. The symmetric key algorithm uses the Xor and shifts, thus the only attack possible to the Blowfish is by the brute-force attack. Contrary to this the domain in which RSA generates the keys is small hence even 1024 bit RSA can be hacked although generally it takes about 2 days. Table 2 and 3 respectively give the encryption and decryption time of RSA and Blowfish (used in present work), while figures 8 and 9 gives the graphical representation of the performance of the same.

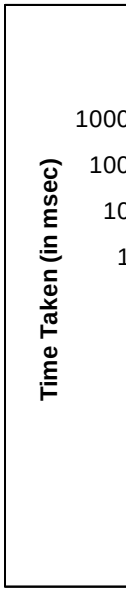
Table –2

File size(in KB)	Encrypt time(in msec)	Decryption time(in msec)
1	30	1429

2	25	2994
3	40	4419
4	54	5768
5	61	6966
10	108	13620
20	191	26991
50	467	67408
100	859	134384
200	1720	267436
500	4285	670648
1000	8721	1385672

Table –3

File size(in KB)	Encrypt time(in msec)	Decryption time(in msec)
1	15.07	12.19
2	24.635	11.24
3	17.2	23.89
4	20.5	13.3
5	14.8	14.6
10	26.2	47.3
20	40.6	41.5
50	87.4	69
100	168.2	122.37
200	326.5	262.1
500	818.5	621.2
1000	1735.04	1251.1
2000	3895.7	2930.6



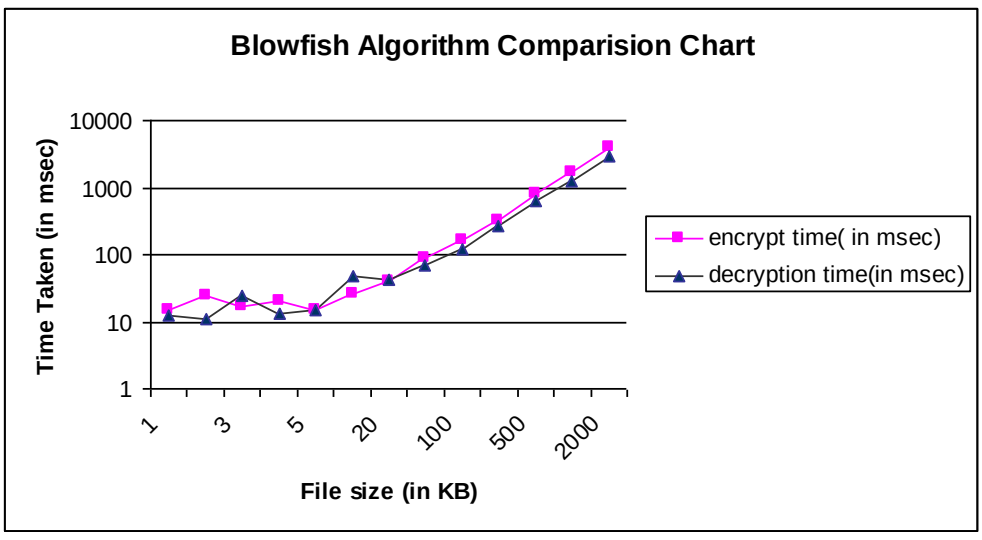


Figure 8 RSA Encryption and decryption performance

Figure 9

The details of Table 2 show that RSA decryption is 156.2547551 times slower than RSA encryption. These results vary depending upon the file size. It's interesting to note that for 2KB file takes total 3019 milliseconds. The decryption / encryption ratio is 119.76, with the initial packet used being less than 2KB. For the same size of the files when Blowfish performance is compared, it's observed that the average decryption to encryption ratio is 0.897610366. That means the encryption and decryption depends only on the file size, which speaks of the advantage of proposed method over the earlier.

Blowfish compared modularly with RSA also produces interesting results. Blowfish encryption is on average 3.907774063 times faster than RSA algorithm. The relevant data is presented in table-4 and graphically represented in figure 10.

Table -4

File Size (KB)	RSA Encryption (msec)	Blowfish encrypt (msec)
1	30	15.07
2	25	24.635
3	40	17.2
4	54	20.5
5	61	14.8
10	108	26.2
20	191	40.6
50	467	87.4
100	859	168.2
200	1720	326.5
500	4285	818.5
1000	8721	1735.04

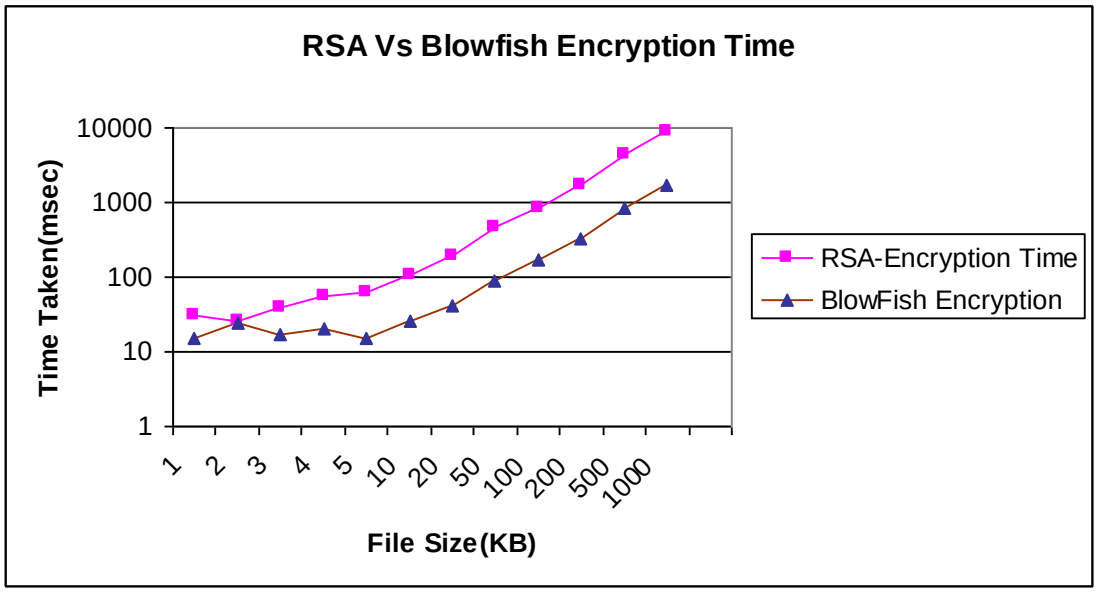


Figure 10

The most interesting feature of all is decryption time of blowfish over RSA. Blowfish is on average 641.694999 times faster than RSA. The relevant data regarding the same are presented in table 5 and graphically represented in fig 11. Such a huge difference in decryption time is due to the arithmetic operations that load the machine. Although RSA is popular and used widely in PKI still the data security is at the stack due to the mathematical solutions and increasing speed of the processors. The process of transmitting the private key is still unknown.

Table -5

File Size(KB)	RSA Decryption(msec)	Blowfish Decrypt (msec)
1	1429	12.19
2	2994	11.24
3	4419	23.89
4	5768	13.3
5	6966	14.6
10	13620	47.3
20	26991	41.5
50	67408	69
100	134384	122.37
200	267436	262.1
500	670648	621.2
1000	1385672	1251.1

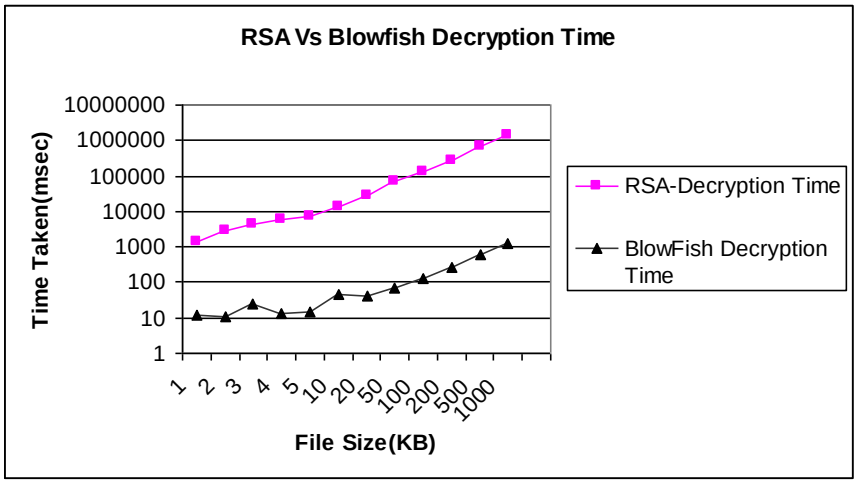


Figure 11 RSA V/S Blowfish Decryption

In the present method the keys are made disposable and the graph warns out in about 2 to 2.5 hours, in the present work it being 143.225 minutes. The relevant data of the same is given in table-6. This makes impossible to compromise the key value within this period. The graph in figure 12 shows the details of the performance for various values of decay factor. The acknowledgement values are checked up to an accuracy of 10^{-7} leading to a better key management system.

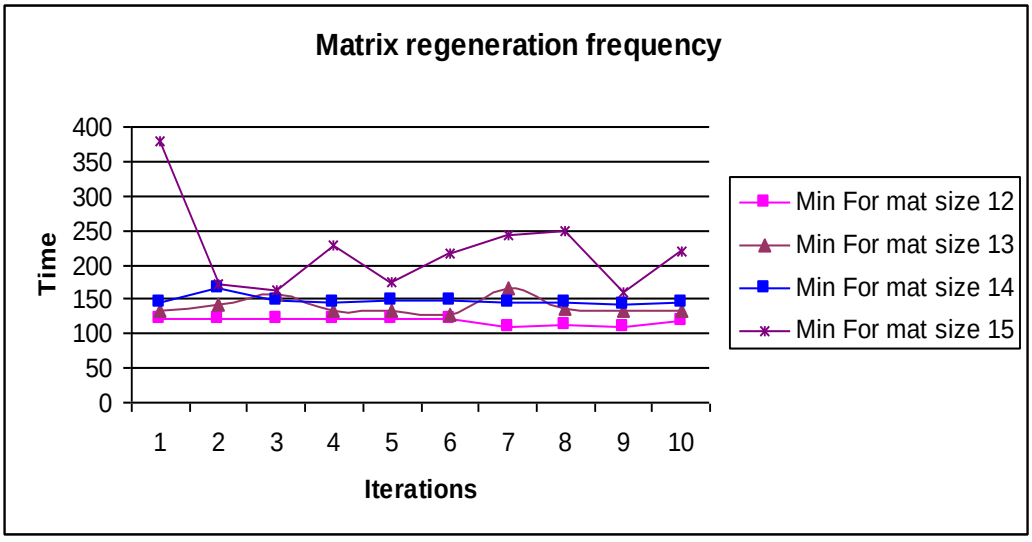


Figure 12 Graph Regeneration frequencies

4.0 Conclusion

In conclusion it may be said the presented system of key management apart from being faster has all the security features. The efficiency in time management in all the aspects namely Key Generation, key Transmission, Key Revocation , Encryption and decryption is found to be much better than the widely used PKI. This speed do not compromise the security aspects of the system in fact this efficient time management has helped to build up the secured transmission.

Acknowledgment: The author Santosh Deshpande acknowledge the help and encouragement shown by their respective principals and managements in carrying out this work.

References:

[1] Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference on Volume: 1, 5-7 April 2004 Pages:17 - 21 Vol.1

[2] Ron Rivest speech at the Cambridge University
[3] Security pitfalls in cryptography by Schneier 1998 Counterplane systems.
[4] Parallel Implementation of An Ant Colony Optimization Metaheuristic With OPENMP*Pierre Delisl, Michaël Krajecki, Marc Gravel, Caroline Gagné*
[5] Applied Cryptography *by Bruce Schinner 2nd Edition*
[6] Two Formal Views of Authenticated Group Diffie-Hellman Key Exchange by E. Bresson, O. Chevassut2,3, O. Pereira, D. Pointcheval and J.J. Quisquater February 28, 2002
[7] E. Bresson, O. Chevassut, and D. Pointcheval. Provably authenticated group Diffie- Hellman key exchange - the dynamic case. In C. Boyd,editor, Advances in Cryptology - Proceedings of AsiaCrypt 2001, pages 290–309. LNCS Vol. 2248, 2001.
[8] E. Bresson, O. Chevassut, D. Pointcheval, and J-J Quisquater. Provably authenticated group Diffie-Hellman key exchange. In P. Samarati, editor, Proceedings of the 8th ACM Conference on Computer and Communications Security, pages 255–264. ACM Press, 2001.
[9] Dynamic management of IPSec parameters by Ghislaine 26-29 October 1999.
[10] Ten Risks of PKI: What You're not Being Told about Public Key Infrastructure By Carl Ellison and Bruce Schneier.
[11] NIST Special Publication 800-25 Computer Security Computer Security Division Information Technology Laboratory National Institute of Standards and Technology Gaithersburg, MD 20899-8930 October 2000.
[12] Public Key Distribution through "cryptoIDs Proceedings of the 2003 workshop on New Security Paradigms. Originally presented at NSPW 2003.
[14] M. Fischlin, The Cramer-Shoup Strong-RSA signature scheme revisited," in *Public Key Cryptography -PKC' 03*, LNCS 2567, pp. 116-129, Springer, 2003.