Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Engineering 38 (2012) 1432 – 1443

Procedia Engineering

www.elsevier.com/locate/procedia

International Conference on Modeling, Optimisation and Computing (ICMOC 2012)

Hierarchical key management scheme for securing mobile agents with optimal computation time

P.Vijayakumar[a,]*, K.Anand[b], S.Bose[a], V.Maheswari[c], R.Kowsalya[c], A.Kannan[a]

[a]Department of Computer Science and Engineering, Anna University, Chennai,TamilNadu-600 025, India.
[b]School of Information and Communication Technology, KTH, Royal Institute of Technology, Stockholm, Sweden.
[c]Department of Computer Science and Engineering, University College of Engineering Tindivanam, Villupuram, India.

Abstract

The Mobile agents are well suited for accessing contents of distributed web based applications including internet business. Such kinds of access must be secure enough enabling only authorized mobile agents to retrieve the contents from the distributed servers. This kind of security is provided by several algorithms including RSA based digital signature and elliptic curve digital signature cryptography. The main differences among the algorithms of the past literatures lie in the computational time of encryption and decryption of the keys used for encrypting and decrypting the content. In this paper we propose a modified hierarchical date constrained key management scheme which potentially reduces the overall computation time needed for key derivation and key signature check operations. Different algorithms are compared with our proposed mechanism and the experimental results show that the proposed method reduces the overall key derivation and key signature check computational time.

Keywords: Mobile Agents, Distributed Web Applications, Elliptic Curve Digital Signature, Computation Time, Access Control.

1. Introduction

In recent days, secure transmission of information from one host to another in an open network environment is a challenging issue. At the same time, large number of users accessing the information over the open network increases the network delay and load.

*Corresponding author. Tel:91-9940896665.
E-mail address: vijibond2000@gmail.com(P.Vijayakumar).

doi:10.1016/j.proeng.2012.06.177

To resolve these problems, we integrate the Mobile Agent (MA) which can adapt itself to any type of internet environment. Additionally, an access key hierarchy based algorithm is introduced to provide secure access of content in an open distributed network environment.

The access key hierarchy [3], [14] consists of various levels of keys generated by Central Authority assigned for various users based on the user rights. The two main operations performed in an access key hierarchy are user join and leave. Whenever a new user joins or leaves from the content access it is the responsibility of the Group Centre (GC)/Central Authority (CA) to disallow the users to access the content present in the server by renewing the present keys located in the access key hierarchy. To avoid the illegal access of data, we integrate date constraint method in our proposed algorithm. This scheme makes the encryption key valid within the date constraint and there is no need to update the keys. In order to generate and distribute the keys to the authorized users we use the elliptic curve digital signature algorithm.

For accessing the contents distributed in open network internet we use mobile agent [9], [10] which is a software program that can roam freely in an internet environment. Moreover, it has the ability to adapt itself to various types of online servers and detects the environment autonomously. This mobile agent program is created and used for self-managing, self-controlling and self-resolving as it has the ability to control its behavior and status. Elliptic curve cryptography is used to protect the mobile agent from various types of security threats. ECC enhances the security of mobile agent and keeps the mobile agent away from malicious attacks. The remainder of this paper is organized as follows: Section 2 provides the features of some of the related works. Section 3 describes the overall system architecture and structure of the mobile agents. Section 4 discusses the proposed key derivation protocol and a detailed explanation of the proposed work. Section 5 analyzes the comparative performance of our proposed algorithm with the other existing key derivation and key signature check methods. Section 6 gives concluding remarks and suggests some future directions.

2. Literature survey

Volker and Mehrdad [1] proposed a key management and access control method for mobile agents. They introduced a tree based mobile agent structure which supports authentication, security management and access control. Using mobile agents, the content of the data are transmitted from one host to another. For protecting the content of the mobile agent, encryption and digital signature algorithms are used. Encryption is used to achieve the confidentiality and digital signature is used to achieve the integrity and authenticity. The main limitation of this paper is that large amount of storage is required for storing the secret keys.

A mobile agent framework called Secure and Open Mobile Agent (SOMA) proposed in [2] is used to enhance the performance of Volker's approach. In that approach, the decryption keys are repeatedly used, thus increasing the public key computation time. For improving this approach two novel methods are presented. The first method is used to design the top down approach and the second method is used to reduce the size of the public parameter. In the first scheme, the leaf nodes are considered as secret keys and the remaining nodes are considered as super keys which are used to derive the secret keys. In this scheme, the authorized host only derives the secret keys that are used to encrypt the confidential files. The limitation of this paper is that the size of the public parameters will grow when the number of visiting hosts and confidential files are increased and everyone derives the secret key if he/she knows super key. Hence it makes the system less secure.

The time bound hierarchical key management scheme proposed in [6] is used to allow users to access data only within the valid time period. It also explains the characteristics and structure of the mobile agents and also discusses the security threats faced by mobile agents. It overcomes the drawbacks of the

Volker and Mehrdad method. Bilinear pairing method is introduced to integrate the time bound key management scheme in which the users can decrypt the confidential document only within the valid time period. Hence it achieves more security. The drawback of this approach is that the bilinear pairing is very expensive in terms of the computation time.

An id-based remote mutual authentication with key agreement scheme [5] was proposed for mobile devices on elliptic curve cryptosystem. The main advantage of this scheme is that it provides mutual authentication and supports session key agreement between user and the server. This scheme is more efficient compared with time bound, because there is no need to compute the public keys. In addition, the users and the server do not need to perform additional computations for verifying the other party's certificates and it provides efficiency. But the system spends more on computation to keep the key updated.

Tzen-Long Chen proposed an efficient date-constraint hierarchical key management scheme for mobile agents [11]. In this approach, every individual user has a separate private key and a date is attached to it. The key is valid within the date only and there is no need to update the key that reduces the computation time. The main limitation of this approach is that it increases the computation time for both key derivation and key signature check phase. This computation complexity can be reduced by reducing the number of point multiplication used in the key derivation and signature phase. Therefore, in this paper we propose an elliptic curve digital signature based new key derivation and signature algorithm that reduces computation time by reducing the number of point multiplication used in the existing approaches.

3. Overall system Architecture

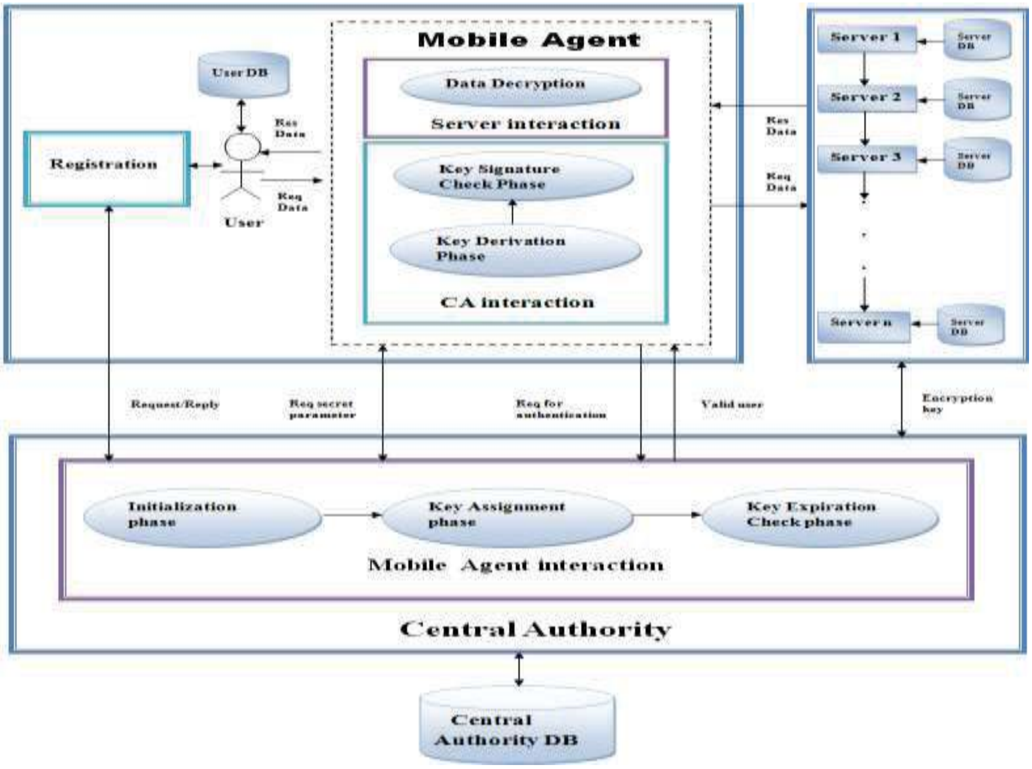


Fig. 1 Architecture of proposed key management scheme

The overall architecture consists of three components. They are Authorized Users (AU), CA (Central Authority) and Server. Authorized Users are the internet users who are allowed to access various files located from various servers in the distributed/internet environment. For accessing the files, searching the files and decrypting the files we use a new component called as mobile agent in the Authorized user's area. Mobile agent is the software program [1] that can roam freely in the Internet environment from local host to other remote hosts in a network and executes tasks assigned by its user. Today mobile agents are not only used for distributed computation and data search at remote environments, it is also used in network management and workflow system. AU first completes registration process with CA and CA assigns the private key to each user during the registration process. Then user sends the file request to the server through the mobile agent. The file request contains file name and ID of the user who is sending the request. A mobile agent submits the ID of the user and the file names to be accessed which was obtained from AU to CA. Then CA checks whether the user is authorized user or not by using the various phases. If the user is an AU then the CA derives the corresponding private key of user from the users ID. Using this private key and base point selected from the elliptic curve, the public key is calculated. CA also checks whether the key is expired or not using the date constraint in key expiration check phase. If the current date is valid, then the request sends to the signature creation phase. Then the CA chooses six random numbers for calculating expired date of the user using one way hash chain function in key assignment phase. Finally CA sends the signature and some parameters to MA for signature verification and to compute a secret number. MA computes the secret random number, using which it also derives its subsequent level keys in the key derivation phase. Finally MA computes decryption keys of each file and the file is decrypted using this decryption key.

3.1 Frame work of mobile agent structure

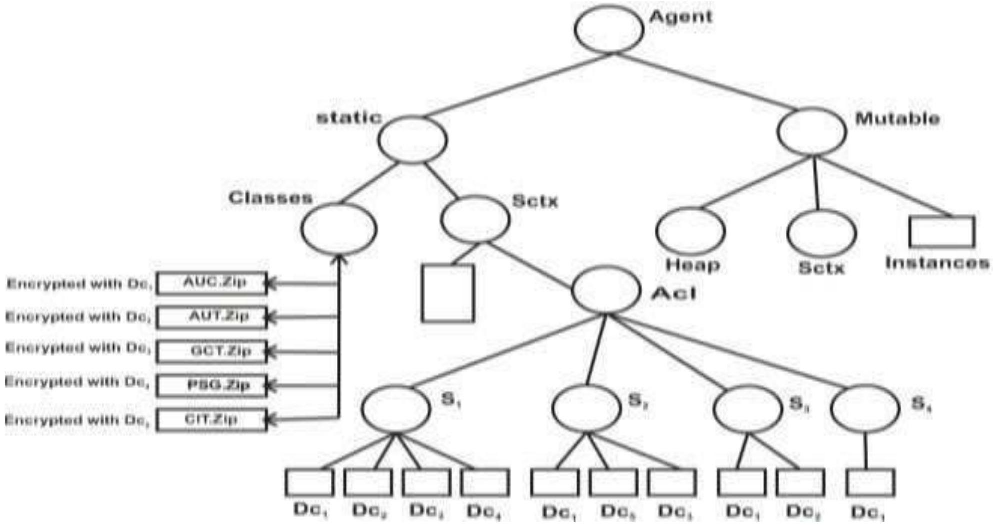


Fig.2 Access control and key management

The mobile agent structure is represented as two parts namely static and mutable parts [1], [2], [4]. The static part of the agent stores all the static information's that are not going to be changed during the agent's lifetime. This information includes security policy, group classification, certificates and access control keys. To prevent mobile agents from malicious attack by hosts, the server will use digital signature on the static part to ensure the completeness and authenticity of data. The mutable part stores all the dynamic information that is gathered by a mobile agent. For instance, the status or information collected by the agent can be modified by the visited hosts on the network. Therefore, each visited host must digitally sign for the information that has been modified.

The Fig.2 shows the mobile agents structure and its components such as classes, Sctx, Acl etc. In this structure there are five zipped files: AUC.Zip, AUT.Zip, GCT.Zip, PSG.Zip and CIT.Zip. As indicated in the Fig.2, the files are encrypted by Dc_1, Dc_2, Dc_3, Dc_4 and Dc_5 respectively. On the right side of static part, the nodes S_1, S_2, S_3 and S_4 represents different servers under access control keys. If these servers are authorized to access the specified files, the decryption keys will be copied to the corresponding server folders. S_1 is granted access to all the encrypted files. If the keys ($Dc_1/Dc_2/Dc_3/Dc_4$) are copied to the folder of S_1. Based upon the figure's decryption, S_2 is admitted to access the three encrypted files AUC.Zip, CIT.Zip and GCT.Zip. Therefore, the keys ($Dc_1/Dc_5/Dc_3$) will be duplicated to the folder of S_2. In addition, S_3 is only allowed to access the two encrypted files AUC.Zip and AUT.Zip accordingly, only the keys Dc_1 and Dc_2 are in the folder of S_3. Finally, S_4 is permitted to access the first encrypted zipped file, AUC.Zip and then it will just have Dc_1 for accessing the designated file.

3.2 Structure of decryption keys for mobile agents

The following tree structure is used to explain how to access leaf nodes confidential file in a hierarchical structure. Consider for example *Filej* is the encrypted confidential file and C_i is the internal node which also represents a user. pr_i represents the secret key/private key held by the user. When pr_i has permission to access some encrypted files, it can obtain confidential file from their corresponding leaf nodes. In this structure taking node C_2 as example, C_2 holds secret key pr_2 and based on its access right, it can access File1, File2, File3, File4 and File5. However, the node C_2 is not permitted to access File6 as the node C_2 does not have access rights to access the File6.

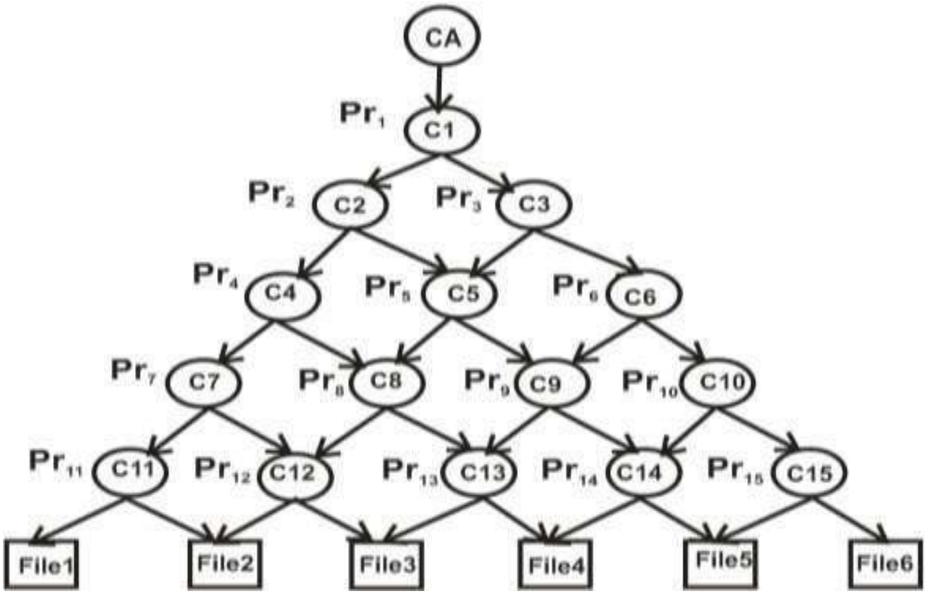


Fig.3 Structure of decryption keys for mobile agent

4. The Proposed Key derivation protocol

4.1 Elliptic curve cryptography

Elliptic Curve Cryptography (ECC) is an approach to public key cryptography [13], [15], [16] based on the algebraic structure of elliptic curves over finite fields. In 1980s, Miller and Koblitz introduced elliptic curves into cryptography and Lenstra showed how to use elliptic curves to factor integers. The ECC is better than other public key cryptosystems in terms of having lower memory, computational requirements, higher security and faster implementations. The main advantage of ECC is its small key size as 160 bit which is considered as secured as 1024 bit key in RSA. Two families of elliptic curves are prime curves and binary curves. Prime curves defined over Z_p and binary curves constructed over $\left(GF(2^n)\right)$. Prime curves are suitable for software application, because it does not require extended bit-fiddling operation. Binary curves are suitable for hardware application because it requires a few logic gates to build an efficient and fast cryptosystems. The elliptic curve over Z_p, defined modulo a prime p, is the set of solutions (x, y) to the equation is $E_p(a,b)\colon y^2 = x^3 + ax + b(mod\,p)$, where $(a,b) \in Z_p$ and $4a^3 + 27b^2\ mod\ p \neq 0$.The condition $4a^3 + 27b^2\ mod\ p \neq 0$ is necessary to ensure that $\mathrm{y}^2 = \mathrm{x}^3 + \mathrm{ax} + \mathrm{b}(\mathrm{mod\ p})$ has no repeated factors, which means that a finite abelian group can be defined based on the set $E_p(a,b)$. The definition of an elliptic curve also includes a point at infinity denoted as O which is also called the zero point. The point at infinity O is the third point of intersection of any straight line with the curve, so that there are points including(x, y), $(x, -y)$ and O on the straight line. Addition operation has been used over $E_p(a,b)$. The addition rules are given below.

(1) $+P = P$ and $\mathrm{P} + \mathrm{O} = \mathrm{P}$, where O serves as the additive identity.
(2) $-O = O$.
(3) $P + (-P) = (-P) + P = O$, where $-P$ is the negative point of P.
(4) $(P + Q) + R = P + (Q + R)$.
(5) $P + Q = Q + R$.

For any two points $\mathrm{P} = \left(\mathrm{x_p}, \mathrm{y_p}\right)$ and $\mathrm{Q} = \left(\mathrm{x_q}, \mathrm{y_q}\right)$ over $E_p(a,b)$, the elliptic curve addition operation and which is denoted as $\mathrm{R} = \mathrm{P} + \mathrm{Q} = (\mathrm{x_r}, \mathrm{y_r})$ satisfies the following rules:

$$x_r = (\lambda^2 - x_p - x_q)\ mod\ p,\ \ y_r = \left(\lambda\left(x_p - x_r\right) - y_p\right)\ mod\ p$$

$$\text{Where } \lambda = \left\{ \begin{matrix} \left(\frac{y_q - y_p}{x_q - x_p}\right)\ mod\ p, if\ P \neq 0 \\ \left(\frac{3x_p^2 + a}{2y_p}\right) mod\ p\ , if\ p = 0 \end{matrix} \right\}$$

4.2 Key Derivation Protocol

The proposed key derivation protocol uses elliptic curve digital signature algorithm [8], [12] for providing the security in CA's and MA's area. The following steps explain public key calculation, signature generation, key derivation, key expiration check and signature verification phases in detail.

Step 1: CA assigns a private key to each user and defines the elliptic curve $E_p(a,b)$, where $y^2 = x^3 + ax + b\ (mod\ p)$ among which a and b must satisfy $4a^3 + 27b^2 \neq 0\ (mod\ p)$ and p is a large prime number.

Step 2: Choose a base point $\mu = (x, y)$ on $E_p(a,b)$, and then user calculates the public key using

$$pu_i = pr_i\,\mu\ \ mod\ p$$

$$pu_j = pr_j\, \mu \, mod\, p$$

Step 3: Next CA randomly chooses six random numbers $\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5$ and γ_6.Then generates $\beta = \beta_1 \parallel \beta_2 \parallel \beta_3 \parallel \beta_4 \parallel \beta_5 \parallel \beta_6$ where $\beta_1 = H^y(\gamma_1)$, $\beta_2 = H^{100-y}(\gamma_2)$, $\beta_3 = H^m(\gamma_3)$, $\beta_4 = H^{12-m}(\gamma_4)$, $\beta_5 = H^d(\gamma_5)$ and $\beta_6 = H^{31-d}(\gamma_6)$. Where y, m and d represents year, month and date respectively, of the date of expiry and " ‖ "is the concatenation operator.

Step 4: Calculate $\alpha = H^{100}(\gamma_1) \parallel H^{100}(\gamma_2) \parallel H^{12}(\gamma_3) \parallel H^{12}(\gamma_4) \parallel H^{31}(\gamma_5) \parallel H^{31}(\gamma_6)$.

Step 5: CA chooses another secret random number λ , and uses the previously calculated parameters α and β to calculate signature date-bound warrant $\alpha \oplus \beta$ and also the following public parameters σ, Z, M and the private key pr_j of user C_j. The calculations are as follows:

$$M = \lambda * \mu = (x_1, y_1)\ and$$
$$\sigma = x_1\, mod\, p$$
$$Z = \lambda^{-1} * (H(\alpha \oplus \beta) + pr_i * \sigma)\, mod\, p,$$ pr_i is the private key of C_i
$$pr_j = H(\lambda, ID_j)$$ Where ID_j is the public identifier of C_j.

Step 6: If $C_i \geq C_j$, then C_i can obtain C_j's private key in the following manner:

Use public parameter σ, Z, α and β to calculate secret parameter λ , as follows:

$$\lambda = Z^{-1} * (H(\alpha \oplus \beta) + pr_i * \sigma)\, mod\, p,$$ pr_i is the private key of C_i.

Step 7: Calculate C_j's private key $pr_j = H(\lambda, ID_j)$.

Step 8: Then calculate $\alpha' = H^{100-y}(\beta_1) \parallel H^y(\beta_2) \parallel H^{12-m}(\beta_3) \parallel H^m(\beta_4) \parallel H^{31-d}(\beta_5) \parallel H^d(\beta_6)$. If α' is not equal to α, then it means the key is already expired.

Step 9: Using the following equation, key signature check is completed by mobile agent.

$$k = Z^{-1}\, mod\, p$$
$$G_1 = k * H(\alpha \oplus \beta)\, mod\, p$$
$$G_2 = (\sigma * k)\, mod\, p$$
$$V = (G_1 + G_2 * pr_i)\mu = (x_1, y_1)$$

Calculate, $\omega = x_1 mod\, p$

$$M = \lambda * \mu = (x_1, y_1) \text{ and}$$
$$\sigma = x_1\, mod\, p$$

Step 10: Judge whether ω is equal to σ. If the two are equal then the signature is true.

Mathematical Proof:

$$\begin{aligned} V &= (G_1 + G_2 * pr_i)\mu = (x_1, y_1) \\ &= \Big(\big(k * H(\alpha \oplus \beta)\big) + (\sigma * k * pr_i)\Big)\mu \\ &= (k * (H(\alpha \oplus \beta) + (\sigma * pr_i)))\mu \\ &= \lambda * \mu \\ &= M \\ &= (x_1, y_1) \end{aligned}$$

Table 1. Computation complexities of various key derivation algorithms

	DCHK	**EDHK**	**Proposed approach**
Central Authority (CA)	$(4Exp + 2Inv + 8Mod + 20H\ + 5M)$	$(5PM + 1Inv + 2Mod + 20H + 2M)$	$(2PM + 1Inv + 3Mod + 20H + 2M\ )$
User	$(2H + 2M + 3Mod + 3Exp)$	$(2H + 2M + 1Mod + 1Inv + 1PA + 2PM)$	$(1H + 5M + 4Mod + 1Inv + 1PM)$

The Table 1 summarizes the overall computation complexity of our proposed Key Derivation protocol with most of the existing well known key derivation protocols DCHK [7] and EDHK [11]. The notations used for comparisons are defined as: Exp defines the number of exponential operations, Inv is the number of inverse operations to be performed in various algorithm by using extended Euclidean algorithms [16], Mod is the modular division operations used in various algorithms, H, M, PM and PA denotes Hashing, Multiplications, Point Multiplications, Point Additions respectively. PM alone takes ‘n’ point doubling operation if λ is equal to an n-bit binary number and it would take (a-1) number of point addition if the value λ consists of ‘a’ number of 1’s. Hence point multiplication takes more computation time than the normal traditional multiplication. However PM takes less computation time when compared with exponential operations. Therefore, proposed approach takes less computation time when it is compared with DCHK since it takes only 2PM, whereas DCHK takes 4 Exp. Similarly our proposed approach takes less computation time compared with EDHK since it takes 2PM whereas EDHK takes 5PM in the CA area. Moreover our proposed approach takes less computation time in the user's side compared with EDHK since it does not use PM in the user's side.

5. Performance Analysis

Table 2. Computation time complexities of various key phases

	DCHK(ns)			**EDHK(ns)**			**Proposed Method(ns)**		
	16 bit	**32 bit**	**64 bit**	**16 bit**	**32 bit**	**64 bit**	**16 bit**	**32 bit**	**64 bit**
Key Initialization	1600124417	1291163383	1777496945	1256696766	1252036943	1546020918	122509119	1252036785	1539056537
Key Assignment	13277579684	16310765178	18439429736	10892414530	14927868579	17259713769	10892414429	12674881216	17074694181
Key Derivation	3400416	4918147	6754401	3295533	4689462	6313585	3131462	4343105	5913585
Key Expiration	27563131	28520380	30561050	19374416	19499124	19654424	19295886	19498114	19653439
Key Signature	28080974	50840465	62443371	22122763	33617340	43734521	11202652	25135795	36659748

The proposed method has been simulated in JAVA (in a P4 machine with 2GB RAM) and we have analyzed the key initialization, assignment, derivation, expiration and check phases computation time with existing approaches to perform the digital signature operation. Table 2 shows the measured computation time in nanoseconds for such comparisons. It is evident from the values that the computation time for our proposed algorithm is found to be better both in the GC and MA area than the other algorithms.

The graphical result shown in Fig.4 and Fig.5 are used to compare the key derivation and key signature check phase computation time of proposed method with the existing methods. It compares the results obtained from our proposed key derivation protocol with the existing RSA based and ECC based approch which are named as DCHK and EDHK. From Fig.4 it is observed that when the key is 64bits, the key derivation time is found to be 5.9136ms in our proposed approach for updating all the keys from the root node to the leaf node, which is better in comparison with the other existing schemes. Moreover, if the number of files to be accessed by a mobile agent increases, then the computation time gradually increases. However, it is less in comparison with the existing approaches. The result shown in Fig.5 is used to compare the key signature check phase computation time of our proposed method with the existing methods. It compares the results obtained from our proposed key check signature phase with the existing approaches and it is observed that when the key size is 64bits, the computation time of key check signature is found to be 36.6577ms in our proposed approach, which is better in comparison with the other existing schemes.

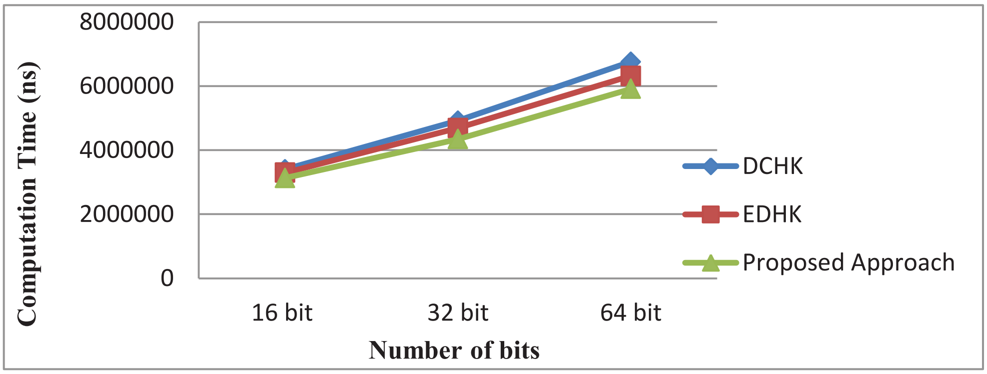


Fig.4. Computation Time for key derivation

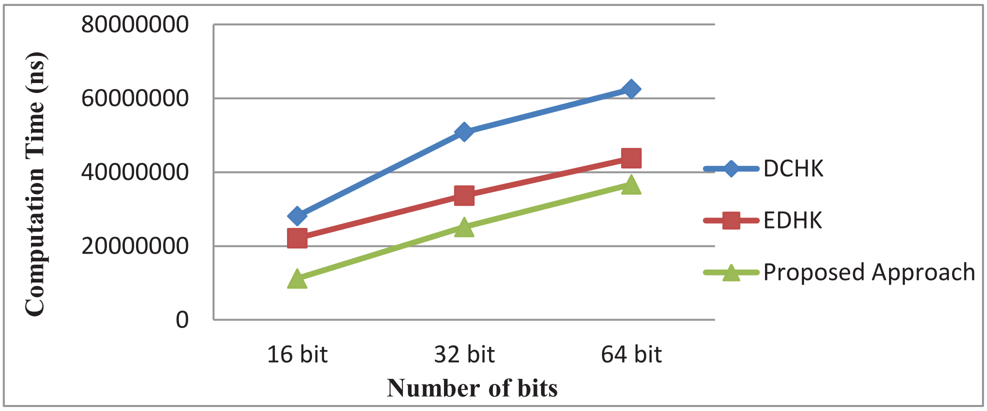
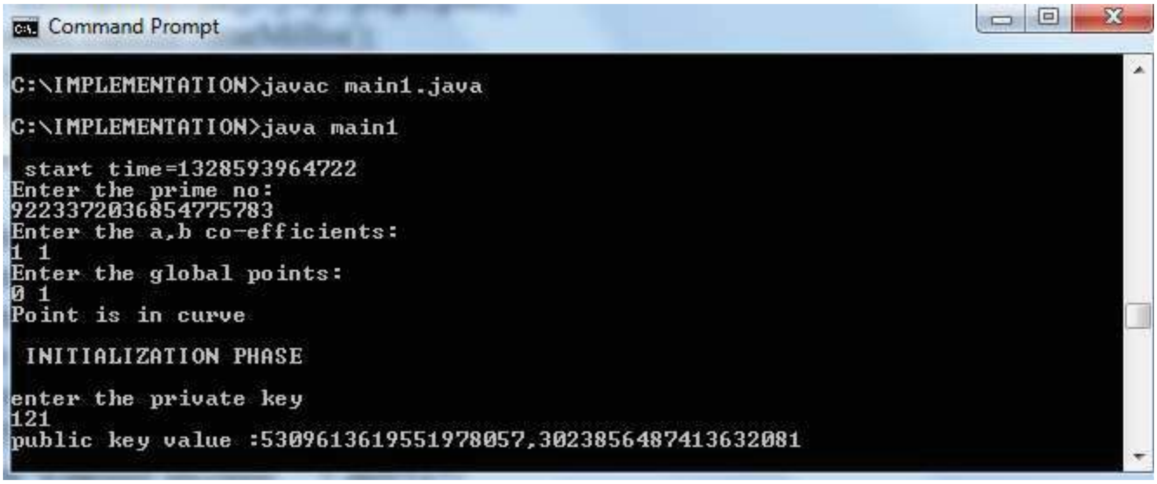


Fig.5. Computation Time for key signature check

The output screen shown in Fig.6, Fig.7 and Fig.8 are used to represent the implemented result of our proposed algorithm.

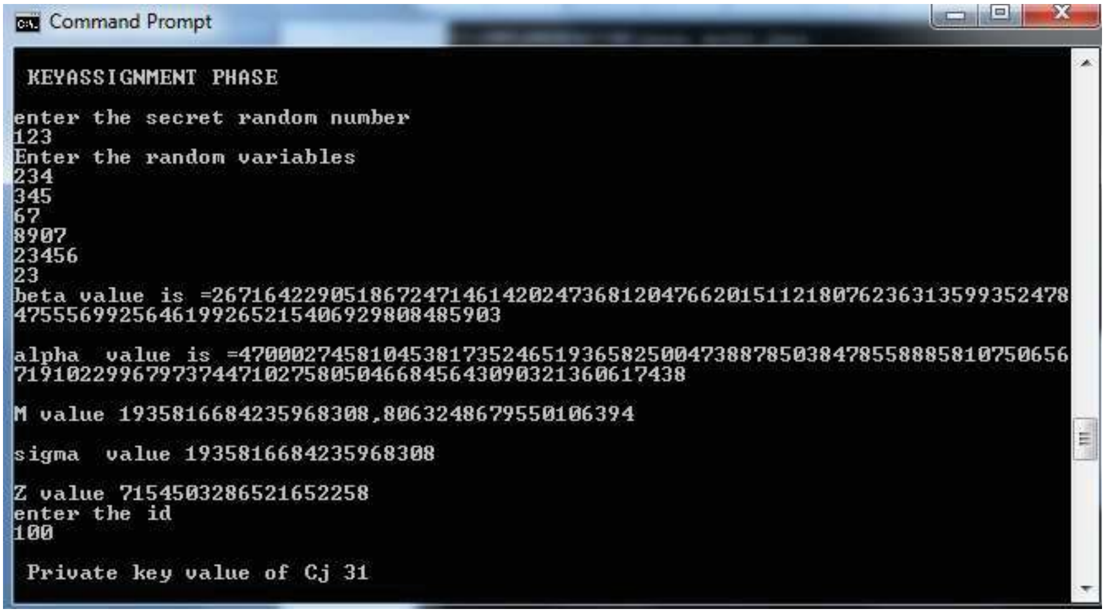
Initialization phase:



```
C:\IMPLEMENTATION>javac main1.java

C:\IMPLEMENTATION>java main1

 start time=1328593964722
Enter the prime no:
9223372036854775783
Enter the a,b co-efficients:
1 1
Enter the global points:
0 1
Point is in curve

 INITIALIZATION PHASE

enter the private key
121
public key value :5309613619551978057,3023856487413632081
```

Fig.6. Public Key Calculation

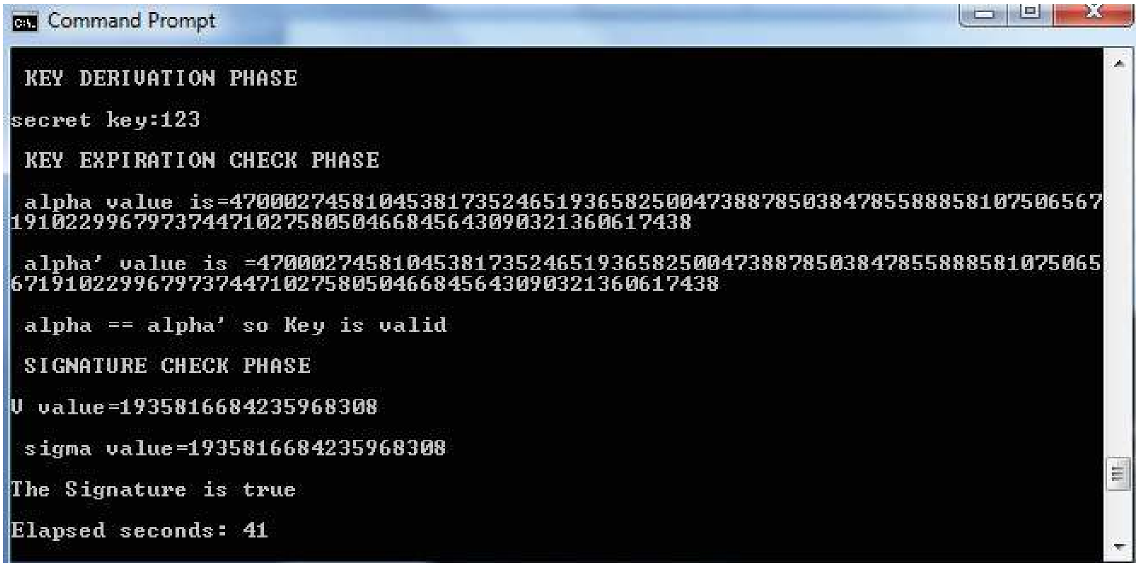
Key Assignment phase:



```
 KEYASSIGNMENT PHASE

enter the secret random number
123
Enter the random variables
234
345
67
8907
23456
23
beta value is =26716422905186724714614202473681204766201511218076236313599352478
4755569925646199265215406929808485903

alpha  value is =470002745810453817352465193658250047388785038478558885810750656
71910229967973744710275805046684564309032136061 7438

M value 1935816684235968308,8063248679550106394

sigma  value 1935816684235968308

Z value 7154503286521652258
enter the id
100

 Private key value of Cj 31
```

Fig.7. Private Key Derivation

Key Derivation and Signature Verification phase:



```
 KEY DERIVATION PHASE

secret key:123

 KEY EXPIRATION CHECK PHASE

 alpha value is=4700027458104538173524651936582500473887850384785588858107506567
191022996797374471027580504668456430903213606174388

 alpha' value is =4700027458104538173524651936582500473887850384785588858107506 5
67191022996797374471027580504668456430903213606174388

 alpha == alpha' so Key is valid

 SIGNATURE CHECK PHASE

U value=1935816684235968308

 sigma value=1935816684235968308

The Signature is true

Elapsed seconds: 41
```

Fig.8. Key Derivation and Signature Verification

6. Conclusion

The mobile agents are not bound to the system it begins execution, unlike the stationary agents. Created in one execution environment, it can transport its state to another execution environment in the network, and resume its execution in the new environment. By authenticating and authorizing a MA, the overall efficiency and security of the system can be improved. In this paper, we propose hierarchical date-constraint key management scheme, to check whether the given key is valid or not. If the key is expired, the user cannot access any information from the server, making key management in the system more efficient. Besides, Elliptic curve cryptosystem is used to provide security to the mobile agents, CA and to the server to reduce the access space of keys and lower the key derivation calculations. The result shows that the proposed scheme is feasible and can be applied in the internet environment and it is not easy for an attacker to hack the data. Thus our proposed scheme can protect the data which is transmitted through the Mobile Agent. Our future work is to devise an idea for developing a new key signature algorithm for providing secure mobile communication in a distributed environment.

References

[1] Volker Roth &Mehrdad Jalali-Sohi. Access Control and Key Management for Mobile Agents. Computer Graphics; 1998, 22 (4), 457–461.

[2] Iuon-Chang Lin, Hsia-Hung Ou, Min-Shiang Hwang. Efficient access control and key management schemes for mobile agents. Computer Standards and Interfaces; 2004, 26 (5), 423–433.

[3] Yu Fang Chung, Hsiu Hui Lee, Feipei Lai, Tzer Shyong Chen. Access control in user hierarchy based on elliptic curve cryptosystem. Information Sciences; 2008, 178(1), 230-243.

[4] Kuo-Hsuan Huang, Yu-Fang Chung, Chia-Hui Liu, Feipei Lai, Tzer-Shyong Chen. Efficient migration for mobile computing in distributed networks. Computer Standards & Interfaces; 2009, 31 (1) 40-47.

[5] Jen-Ho Yang, Chin-Chen Chang. An ID-based remote mutual authentication with key agreement scheme for mobile devices on elliptic curve cryptosystem. Computers & Security.2009,28, 138-143.

[6] Chia-Hui Liu, Yu-Fang Chung, Jin-De Jhuo, Tzer-Shyong Chen, Sheng-De Wang .A Novel Time-bound Hierarchical Key Assignment Scheme for Mobile Agent. IMECS 2010 Vol I

[7] Li, J.H .Hierarchy-based key assignment scheme with date-constraint. Master Thesis, Feng Chia University, Taichung. 2004.

[8] Don Johnson, Alfred Menezes, & Scott Vanstone. The elliptic curve digital signature algorithm (ECDSA). Information Security, 2001, 1, 36–63.

[9] Karmouch, A. Mobile software agents for telecommunications. Guest Editorial, IEEE Communications Magazine, 1998, 36 (7), 24–25.

[10] Lange, D.B., & Oshima, M. Programming and deploying Java mobile agents with aglets. Massachusetts, USA: Addison-Wesley Press.1998

[11] Tzer-Long Chen, Yu-Fang Chung, Frank Y.S. Lin. An efficient date-constraint hierarchical key management scheme for mobile agents. Experts systems with Applications. 2010, 37, 7721-7728.

[12] Aqeel Khalique, Kuldip Singh, Sandeep Sood. Implementation of Elliptic Curve Digital Signature Algorithm. International Journal of Computer Applications (0975 – 8887).2(2), May 2010.

[13] Koblitz, N. Elliptic curve cryptosystems. Mathematics of Computation 1987, 48, 203-209.

[14] Mikhail J.Atallah, Keith B. Frikken, and Marina Blanton. Dynamic and Efficient Key Management for Access Hierarchies. ACM Trans. Inf. Syst. Secure, 12(3),2009.

[15] Wu, S.T. Authentication and group secure communications using elliptic curve cryptography. Doctoral Dissertation, National Taiwan University of Science and Technology, Taipei.2005.

[16] Wade Trappe, Lawrence C. Washington, Introduction to cryptography with coding theory, Pearson Education, Second Edition, (2007) 66-70.