JOURNAL OF AL AZHAR UNIVERSITY ENGINEERING SECTOR
VOL. 12, NO. 42, JANUARY, 2017, 131-143

AN EFFICIENT DISTRIBUTED KEY GENERATION PROTOCOL FOR TRANSIENT CLOUDS

Mona Samir Lackousha, Hisham Dahshan, and Nabil Hamdy Shaker
Science and Technology Center of Excellence, Military Technical College, and Misr International University

ABSTRACT
Cloud computing presents a new approach to allow the development of dynamic, distributed and highly scalable software (e.g., networks, servers, storage, applications, and services) that can be rapidly supplied and allowed us with minimal management effort or service provider interaction. Cloud computing provides many advantages to its customers but failing to solve information security concerns. Strong encryption with key management is one of the core mechanisms that Cloud Computing systems should use to secure data, so key management is an important technology that can help secure applications and data in the cloud. In this paper, a proposed elliptic curve based distributed key generation (ECDKG) scheme for Transient Clouds is presented. Performance analysis showed that this scheme not only has strong security, but also has less computational and communication costs.

KEYWORDS: :**Cloud Computing, Authentication, Encryption, Cryptographic Key Management (CKM), Elliptic Curve Cryptosystem (ECC).Mobile Cloud Computing (MCC), Mobile Services.**

1. INTRODUCTION
Cloud computing is a type of Internet-based computing, and it is one of the foundations of the next generation of computing. Figure 1 shows visual model of cloud computing definition and this model is composed of five essential characteristics, three service models, and four deployment models. Cloud computing provides several benefits to its consumer such as availability, flexible cost model, on demand self services, elastic resources, etc. It facilitates their consumers by providing software, platform and infrastructure as service. Consumers can easily use these services any time everywhere through internet [1, 2]. Clouds are classified into four models based on their infrastructure and these are distinguished by their architecture and functionality: public cloud, private cloud, hybrid cloud and community cloud.

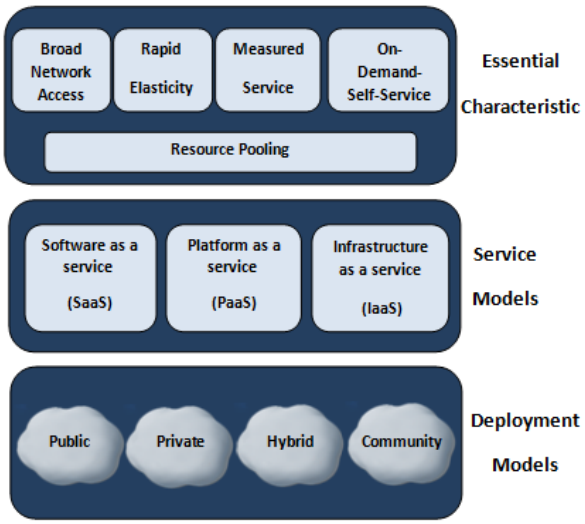


Figure 1: **Cloud computing**

Traditional information security mechanism such as cryptography, digital signature, access control mechanism, and trust management are not sufficient to fulfill the need of information security on cloud paradigm. It also provides suggestion on critical area of cloud computing. Cryptographic key management in cloud computing is a challenge and cryptographic keys should be stored on enterprise domain due to their security. However, cryptographic key management is required at cloud when any application process requires working on plain text at cloud platform and data must access cryptographic key. Cryptographic key management includes all operation that can be performed on cryptographic key except encryption/decryption. These operations comprise but are not limited to generation, revocation, sharing and storage of cryptographic keys.

Extensive research in cloud security includes several techniques for distributed key generation (DKG) protocols are based on either discrete logarithm problem (DLP) over a finite field or integer factorization problem (IFP). Elliptic curve cryptosystems (ECC) are safe against some common algorithmic techniques. Few issues in cryptographic key management for cloud computing is surveyed [3-5] as follows:

The verifiable secret sharing (VSS) scheme is presented in [3]. This scheme uses the dealer (a dealer is defined as the coordinator which distributes the shares to each player). It selects and encrypts a "secret message", s, and gives a "share" of s, to each of n players. All communications use broadcast messages and the players can verify the authenticity of the dealer. Shamir's (n, t, t+1) threshold cryptography can be used as a building block to this VSS.

The first distributed VSS version is presented in [4]. This protocol is based on Feldman VSS with each player acting as a dealer. It specifies n parallel runs of all the players, and each player selects a random secret $z_i \in GF(q)$ and shares it among other players. The player collaboratively constructs a non-disqualified set Q (a non-disqualified set is a set of players who conforms to the protocol and passes all tests) in which the secret is shared.

An improved version of Pedersen distributed VSS is presented in [5] and it is called distributed key generation (DKG).This protocol can afford the attack where the adversary can force the secret key to have a biased distribution. DKG resists to halting players for $n \geq 2t + 1$and t eavesdropping players for $n \geq t + 1$ and t static malicious adversary for $n \geq 3t+1$.Pedersen distributed VSS fails to GJKR [5, 6] attack, and the generated public key does not follow a uniform distribution over the given field.

Transient Clouds [7] is a collaborative computing platform that allows nearby devices to form an ad-hoc network and provide various capabilities as cloud service. Transient Clouds utilize the collective capabilities of the devices present, along with their social and context awareness that cannot be provided efficiently by the traditional clouds.

In Transient Clouds, the Wi-Fi Direct framework has been built into the operating system [8-10]. Wi-Fi Direct is a technology that allows devices to create an ad-hoc network and connect to each other using standard Java sockets. In Wi-Fi Direct, only one device (called the Group Owner) acts as a router, and all of the other client devices that connect to it as shown in figure 2.

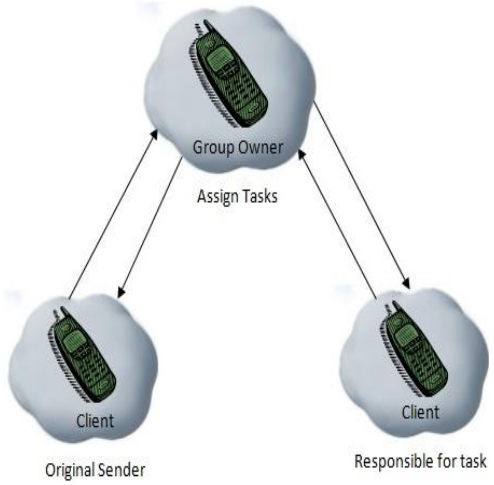


Fig. 2. Transient Cloud

The key management is important issues in multicast communication of Transient Clouds. This paper presented elliptic curve based distributed key generation (ECDKG) scheme, which is based on DF-VSS [5] and uses ECC as the building block. The advantages of using ECC compared to competing approaches are given as follows: 1) Smaller key size for a similar level of secrecy. 2) Much more flexibility with many curves to choose from. 3) More efficient key generation, validation algorithms with a low processing overhead. This paper brought forward a group key management scheme of Transient Clouds based on ellipse curve cipher (ECC) and threshold value technique. This scheme uses threshold value mechanism to boost up the strong quality of the system. Our results show that even in a large network, the sub-second range takes time to generate keys and so, it allows (ECDKG) to be used in Transient Clouds. In the proposed (ECDKG), a timeout mechanism is used to counter measure the halting adversary and the GJKR attack.

The rest of the paper is organized as follows: In Section II, a description of proposed elliptic curve based distributed key generation (ECDKG) scheme for Transient Clouds is presented. Security and performance analysis of the proposed protocol is presented in Section III. Analysis shows that the proposed protocol is based on common adversary models and shielded against these adversaries. Finally, Section IV concludes the paper.

2. DESIGN OF THE PROPOSED GROUP KEY MANAGEMENT SCHEME

A field GF (q) is used as the ground field where q is prime / proper power of prime. A description of this protocol is given below.

A. Group Key Agreement Network Model

According to potential technology of Transient Clouds, this scheme designed a distributed group key agreement network model showed in Figure3.

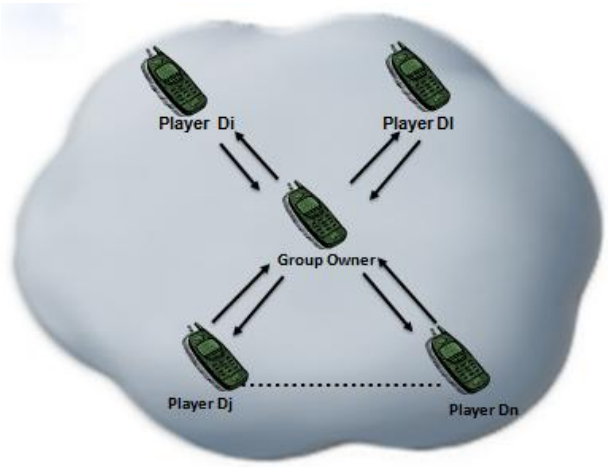


Fig. 3: Distributed group key agreement network model

We assume that there is a private channel, group owner and any player (p_i) can communicate via their respective private channel. We use the Wi-Fi Direct framework, a relatively recent addition to the Android operating system that allows devices to connect to each other directly over Wi-Fi and exchange data. Message broadcasting uses a flooding mechanism, i.e., scoped flooding with a time-to-live scope. In this communication model, we assume that no message loss can occur during transmission.

There are four types of specific adversaries dealt with in this paper:

1- Static malicious adversary. Before the protocol implements, this type of adversaries have already decided which player to mess by all necessary means.

2- Halting adversary. In a protocol execute, a player may not respond to a message either deliberately or due to stop-fail type of failure.

3- Eavesdropper. An adversary passively monitors the channel, and accesses all public messages.

4- Replay adversary. A replay adversary buffers messages and sends these out whenever necessary to impersonate an authorized player.

The design of the proposed protocol has taken into consideration these adversaries in the communication model.

In Transient Cloud Computing, there is one device from set of peer players is elected as the Group Owner (GO), GO is regarded as group member registration authority (RA), and that all other devices must connect to it in order to join the network.

The Group Owner then starts building the new protocol that is based on distributed Feldman verifiable secret sharing (DF-VSS) [6]. This new protocol is called elliptic curve based distributed key generation (ECDKG). Elliptic curve cryptosystems provide a promising alternative with efficiency which is suitable for low-power devices in terms of memory and processing overhead. The new protocol when compared to distributed key generation (DKG) enjoys all its advantages. In addition, flexibility of protocol setup and short key length.

As a result, the proposed protocol uses a complete ad hoc network like network with a star topology. The advantage of star topology, gives the Group Owner full responsibilities for managing key distribution.

B. System Initialization

Supposing that player network includes n players who want to form a secure Transient Cloud Computing. The set of peer players of D_i is $Si = \{Dj \mid j \neq i, 1 \leq j \leq n\}$. Each player has a unique ID= $\{D_1, D2, \ldots, Dn\}$.$D_j \in GF(q)$ and players know these numbers of each other. Each player sends D_j to Group Owner GO. Then Group Owner (GO) broadcasts it to all other players, such that players know these numbers of each other.
The notations of the parameters which are used in the proposed scheme are shown in Table I.

Table I. Notation.

GF*(q)	The induced multiplicative group of GF(q)
G	The main subgroup of order p which is derived from a point T , and used as the base group of ECDKG
Si	The set of peer players of pi, $Si = \{D_j \mid j \neq i, 1 \leq j \leq n\}$
$\oplus$	Elliptic curve point addition by a scalar
$\sum^{\oplus}$	Point summation under point add operation $\oplus$
Θ	Elliptic curve point multiplication by a scalar
$\prod$	Point multiplication under point multiply operation Θ
n	Total number of players in the Transient Cloud Computing
t	Threshold number of players required to provide shares
D_i	Unique identification of each player in a secure group n, $D \in GF^*(q)$
T	Point in E/GF(q), The cardinality of E/GF(q) is a prime number or has a large prime factor for the crypto-graphical purpose. We use p to denote this prime hereafter.
T'	another point in G whose discrete logarithm with respect to T is not known to any of these n players
E/GF(q)	Additive group based on a properly preselected elliptic curve E

In this paper, the point multiplication which is a point multiplied by a scalar in GF*(q) and other point addition are performed in G. the finite field GF (q) is used to perform all arithmetic operation. To calculate Q(x) T, two steps are done; calculate Q(x) in field arithmetic operations in GF (q), second, the modular p is taken (Q(x) mod p).

C. Group Key Generation Protocol

Threshold technique was used in this scheme to generate group key. Each client generates sub-key s_{ij} and uses it to generate group key share GKS. The Group Owner (GO) appears as a trusted third party to enhance the security and efficiency of the network.
The proposed protocol (ECDKG) consists of four algorithms, key distribution (KD), key verification (KV), key revocation (KR) and finally, key generation (KG).
This flow chart in Figure 4 illustrates the scenario of the proposed protocol as follows:

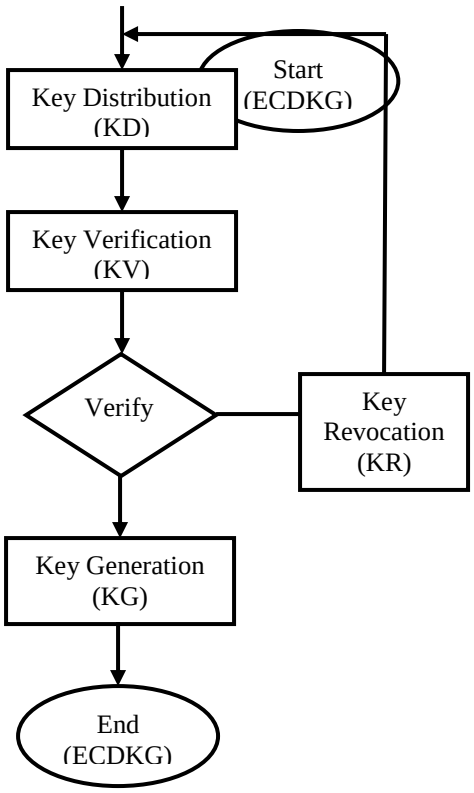


Figure 4: Flow chart of the proposed protocol

Algorithm 1: KD (i, t, T)

1) Player D_i chooses c_{ik}, d_{ik} (k=0, 1, 2,...,t), random numbers uniformly distributed, $c_{ik} \in GF(q)$ and $d_{ik} \in GF(q)$ $(0 \leq k \leq t)$,as polynomial coefficients to establish two secret polynomials $f_i(x)$, $f_i'(x)$ of degree t. Threshold value polynomial function $f_i(x)$, $f_i'(x)$ is shown as follows:

$$\mathbf{f_i(x) = \sum_{k=0}^{t} c_{i\boxdot} x^k} \tag{1}$$

$$\mathbf{f'_i(x) = \sum_{k=0}^{t} d_{ik} x^k} \tag{2}$$

Player D_i computes sub-key share for other members D_j $(j \neq i)$. The computing method is $s_{ij} = f_i(D_j)$, $s_{ij}' = f_i'(D_j)$, $(j \in S_i)$. Then computes:

$$V_{ik} = (c_{ik}T) \oplus (d_{ik}T') \; (0 \leq k \leq t) \tag{3}$$

2) Player D_i sends a message containing s_{ij} and s_{ij}' to Group Owner using the private channel between D_i and Group Owner and then the Group Owner sends s_{ij} and s_{ij}' to p_j using private channel.

3) Player D_i sends a message containing V_{ik} to Group Owner using the private channel between D_i and Group Owner and then Group Owner broadcasts V_{ik} to all other players.

Algorithm 2: KV (D_i, t, T)

Player D_i receives s_{ji} and s_{ji}' sent by Group owner that associated D_j, and then do the following:

1) Verify:

$$(S_{ji}T) \oplus (S_{ji}'T') = \sum_{k=0}^{t \oplus} D_i^{\,k} V_{jk} \quad (4)$$

2) If fails: D_i sends a complaint to Group Owner against D_j. Then Group Owner broadcasts this message to all other players.

3) If D_j receives a complaint to him form Group Owner, D_j sends s_{ji} and s_{ji}' to Group Owner that satisfy the verification equation then Group Owner re-broadcasts them again.

Algorithm 3: KR (D_i, t,Q_j)

$$\sum_{j \in Qi}^{\oplus} s_{j\boxdot}$$

Group Owner Updates share s_i: D_j is removed from Q_i and update s_i= (5)

If one of the following two conditions holds:

1) Group Owner receives t+1 or more distinct complaints against D_j.
2) Received a re-broadcast s_{ji} and s_{ji}', but the received s_{ji} and s_{ji}' still falsifies the verification equation.

Algorithm 4: KG (D_i,Q_i)

Group Owner generates group key share GKS:

1) Each player (D_i) computes its public key, which $c_{i0}$is private key of player(i), as follows:

$$B_{i0} = c_{i0}\, T \quad (6)$$

After computation, player (D_i) sends B_{i0} to Group Owner then Group Owner computes group key share as:

$$GKS = \sum_{i \in Qj}^{\oplus} B_{i\boxdot} \quad (7)$$

Finally, GO broadcasts group key to all players.

3. SECURITY ANALYSIS & PERFORMANCE EVALUATION

In this section, a description of security analysis and performance evaluation of the proposed protocol for Transient Cloud Computing is presented.

1. Security Analysis

In this section, a formal verification of the proposed protocol is presented. The formal verification has been made using the Scyther tool [11, 12]. Scyther tool was developed by CasCremers in 2007 [13]. Scyther, is a formal protocol analysis tool, for the symbolic automatic analysis of the security properties of cryptographic protocols (typically confidentiality or variants of authenticity). It assumes perfect cryptography, which means that an attacker gains no information from an encrypted message unless he knows the decryption key.

Scyther takes as input a role-based description of a protocol in which the intended security properties are specified using claims. Claims are of the form claim (Principal, Claim, Parameter), where Principal is the user's name, Claim is a security property (such as 'secret'), and Parameter is the term for which the security property is checked. The description of a protocol is written in

SPDL (Security Protocol Description Language) language. For the protocol verification, Scyther can be used in three ways:

- *Verification claim*: Scyther verifies or falsifie security properties.
- *Automatic claims*: if user does not specify security properties as claim event, Scyther automatically generates claims and verifies them.
- *Characterization:* each protocol role can be characterized. Scyther analyzes the protocol and provides a finite representation of all traces that contain an execution of the protocol role.

Scyther generates attack graph. This graph presents a way or another where an attack can intrude the system. Ensuring Transient Clouds environment protection means that we should satisfy the following four properties to protect Transient Clouds against different attacks:

Property 1- Confidentiality

This claim is fulfilled if the Group Owner (GO) has the guarantee that all exchanged player data (D_i) is secret. The exchanged user data messages between the player D_i and the Group Owner GO is called Msg. Each information (α) in Msg should remain secret [11, 12, and 13]. The formalization of information confidentiality is given below:

$$\forall \alpha \in \text{Msg}\ (\text{claim}(\text{GO}, \text{Secret}, \alpha)) \quad (8)$$

Property 2- Integrity

This claim is fulfilled if the player D_i and Group Owner GO have the guarantee that all exchanged keys (described as key) are secret and unique. We have included an additional restriction that only claims concerning sessions between trusted agents are evaluated. Its formal definition is shown as follows [11, 12, and 13]:

$$\forall \text{key}(\text{claim}(\text{GO}/D_i\ \text{Secret}, \text{key})) \quad (9)$$

Property 3- Access control

A Transient Clouds should have a correct mechanism to verify that a given user is authorized to use a particular service. A service should always be bound to an authenticated user. Its formal definition is given as follows [11, 12, and 13]:

$$\forall \alpha \in \text{Msg}\ (\text{claim}(\text{GO}, \text{Secret}, \alpha)) \quad (10)$$

Property 4- Freshly of messages

An important part of security protocols is the generation of fresh values which are used for challenge-response mechanisms (often called nonces), or as session keys. This claim is fulfilled if the Group Owner GO and player D_i the guarantee that the session key is fresh [11, 12, and 13]:

$$(\text{claim}(\text{GO}/D_i\ \text{Fresh}, \text{key})) \quad (11)$$

This model is going to be challenged with the following requirements using the Scyther tool.

1. Property 1: In the formal analysis it is proven that an intruder cannot obtain the data exchange between the Group Owner GO and client Di.
2. Property 2: The authorization key exchanged in the authentication protocol is secret.
3. Property 3: The unauthenticated user cannot access the services provided, and cannot impersonate another user. Also, it is not possible to modify the data by an unauthorized individual.
4. Property 4: An adversary cannot obtain the unique Timestamp used in the protocol to prevent replay and man-in-the-middle attack.

Figure 5 show that, the proposed protocol specification describes a set of roles. These roles indicate the behavior of the actual agents in the proposed protocol. There are basic sets used to construct role terms such as: variable, secret and Role [13].There are send events denote the

sending of a message and receive events denote the reading of a message in each role. Every role specification consists of a sequence of events describing the messages the agent shall send and receive, as well as certain security claims. When a protocol is executed, each role can be executed a number of times.

In the Scyther implementation of our proposed protocol, we found that all the claims used to verify the security properties for the proposed protocol with a status (OK) for all the claims and there are no attacks within bounds are found.

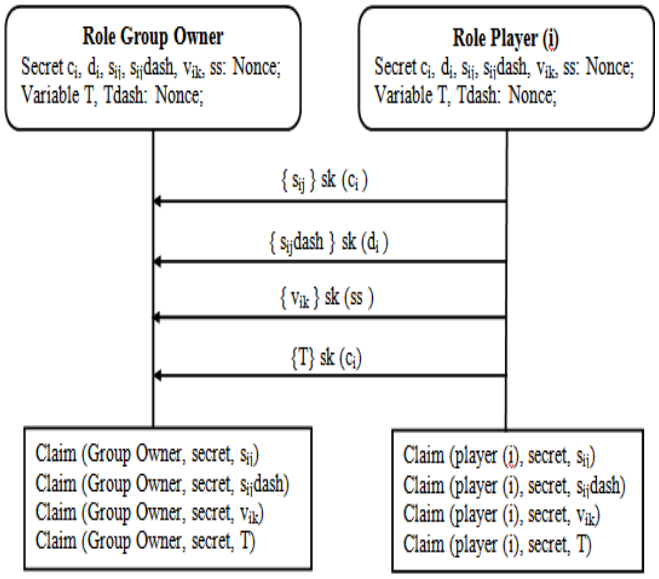


Figure 5: Scyther verifies or falsifiers' security properties for the proposed protocol.

2. Performance Evaluation

The performance evaluation of our proposed threshold key management scheme is presented. We measure the computing cost of our proposed scheme. This highlights the performance improvement achieved in our proposed scheme.

In the proposed scheme, the public key from the key generation protocol follows a uniform distribution in the elliptic curve additive group, and the signature can be generated and verified efficiently. We evaluated the proposed key generation protocol and signature scheme using PARI/GP [14].

The simulation process used the following in its testing environment.

1. Virtual Machine "VMware Player".
2. Pari/gp program.
3. Programming language.

Computing Cost

The computation overhead of the proposed scheme is measured by calculating the time consumed in each algorithm (the key distribution algorithm (KD), the key verification algorithm (KV), the key revocation algorithm (KR) and the key generation algorithm (KG)). The execution time (speed) of the proposed key management scheme indicates its computation overhead such that the higher the computation overhead, the higher the computation timing.

The fundamental operation underlying elliptic curve cryptography (ECC) is point multiplication, which is defined over finite field operations. In elliptic curve cryptography, the point multiplication operation is the most complex operation and the higher the number of point multiplication operations, the higher the required computation timing.

Elliptic curves are defined over either prime fields GF (p) or binary fields GF (2^p). The performance of GF (2^p) can be much improved over GF (p) when using environments in which an arithmetic processor is already available. In the performance evaluation of our proposed scheme, we consider only prime fields GF(p), since binary field arithmetic is in sufficiently supported in PARI/GP [14] and would lead to lower performance.

On a laptop with an Intel® Core i5 Duo 2.4 GHz processor and 4GB memory, PARI/GP [14] is used to evaluate the performance of our proposed scheme. The prime elliptic curve over GF (p) is defined by the equation:

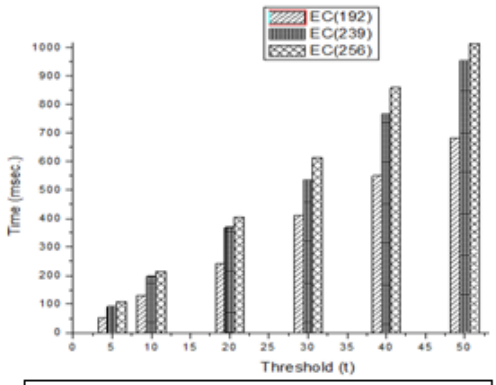
$$Y^2 = (x^3 + ax + b) \bmod p \qquad (12)$$

Where a and b ϵ GF (p) for p is a prime.

The computation overhead of our proposed scheme comes from the fact that the computation in each algorithm which depends mainly on the total number of nodes (n) and the threshold (t).

The following figures (6, 7 and 8) describe the consumed time of each algorithm in the proposed elliptic curve based distributed key generation (ECDKG) in three cases of Elliptic Curve (EC) 192, 239, and 256 for 50 clients and different values of threshold (t).

Figure 6 indicates computing cost of key distribution (KD) for three different elliptic curves as shown:



Number of clients =50			
Threshold(t)	Time (msec.)		
	EC(192)	EC(239)	EC(256)
5	50.4	92	109.6
10	129.6	196.8	215.2
20	242.4	368.8	405.6
30	412	536	614.4
40	549.6	768	859.2
50	682.4	953	1014.4

Figure 6: The Result of Key Distribution Algorithm

The following figure defines the computing cost of key verification (KV) and key revocation (KR) algorithms in the proposed protocol for three cases of elliptic curve.

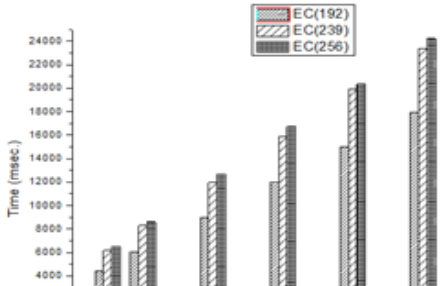
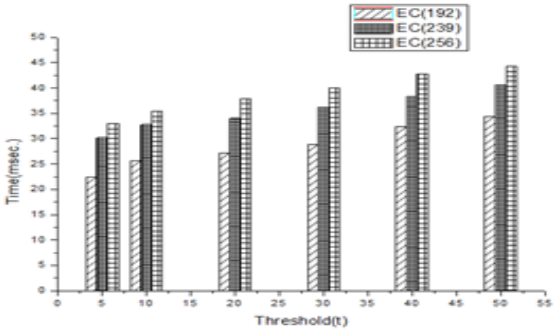


Figure 7: The Result of Key Verification & Key Revocation Algorithms

Finally, this figure illustrates the computing cost of key generation algorithm (KG) in the proposed protocol in the three cases of elliptic curve.



Number of clients =50			
Threshold(t)	Time (msec.)		
	EC(192)	EC(239)	EC(256)
5	22.4	30.2	33
10	25.6	32.8	35.4
20	27.2	34	37.9
30	28.8	36.2	40.1
40	30.4	38.4	42.8
50	32	40.6	44.3

Figure 8: The Result of key Generation Algorithm

The performance of the proposed scheme is evaluated for key sizes: 192, 239, and 256 bits and threshold (t). We note that, time consuming in case of Elliptic curve 192 is the least in all cases of threshold (t).

The following table illustrates the number of point multiplication and addition of each algorithm in the proposed protocol, as these equations in terms of threshold (t), number of point multiplication (M) and number of point addition (A).

Table II Timing Result.

Phase / Number	Key Distribution (KD)	Key verification (KV)	Key Generation (KG)
Number of point multiplication (M) & Number of point addition(A)	t (1A+2 M)	t (2A +4M)	t (1A + 1M)

In key distribution (KD) algorithm, we find that there are one point addition operation and two point multiplication operations and this is obvious in the formula (3).
In key verification algorithm (KV), there are two point addition operations and four point multiplication operations and this is clarify in the formula(4).
In key generation (KG) algorithm, there are one point addition operation and one point multiplication operation as shown in the formula (6), (7).
We note that, time consuming in the key verification algorithm (KV) is more than time consuming in key distribution (KD) algorithm and key generation (KG) algorithm as shown in figures 6, 7 and 8 because number of point multiplication in KV algorithm is more than number of point multiplication in KD and KG algorithms as indicated in table II.

4.. CONCLUSIONS

In this paper, a distribution key generation protocol based on elliptic curve discrete logarithm problem (DLP) is proposed. According to the potential technology of Transient Clouds, this protocol is well suited for Transient Cloud Computing. The proposed protocol provides high level of secrecy with smaller key size as compared to the protocols based on discrete logarithm over finite field. Key generation takes sub-second with practical key length is possible for low power devices and the verification process takes time in millisecond.

REFERENCES

[1] National Institute of Standards and Technology, "The NIST definition of cloud computing," Information Technology Laboratory, 2009.
[2] K. Stanoevska-Slabeva, T. Wozniak, and S. Ristol, "Grid and cloud computing- a business perspective on technology and applications,"Springer-Verlag, Berlin, Heidelberg, 2009.
[3]P. Feldman, "A Practical Scheme for Non-Interactive Verifiable Secret Sharing", Proc. 28th IEEE FOCS, 1987.
[4] T. Pedersen, "A Threshold Cryptosystem Without a trusted Party", Advances in Cryptology – Eurocrypt '91. LNCS 547, Springer-Verlag, 1999.
[5] Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk and TalRabin, "Secure Distributed Key Generation for Discrete-Log Based Cryptosystems", Proceeding Eurocrpt, 1999.
[6] T. Pedersen, "A Threshold Cryptosystem Without a trusted Party",Advances in Cryptology – Eurocrypt '91. LNCS 547, Springer-Verlag, 2013.
[7] M. Satyanarayanan, "Mobile computing: the next decade," in Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond (MCS), June 2010.

[8] M. Gordon, D. Jamshidi, S. Mahlke, Z. Mao, and X. Chen, "COMET:Code Offload by Migrating Execution Transparently," in Proceedings ofOSDI, Hollywood, CA, October 2012.
[9] Y. Zhang, G. Huang, X. Liu, W. Zhang, H. Mei, and S. Yang, "Refactoring Android Java Code For On-Demand Computation Offloading,"ACM Special Interest Group on Programming Languages (SIGPLAN)Notices, vol. 47, no. 10, pp. 233–248, 2012.
[10] R. Kemp, N. Palmer, T. Kielmann, and H. Bal, "Cuckoo: A Computation Offloading Framework for Smartphones," in Mobile Computing, Applications, and Services, ser. Lecture Notes of the Institute for ComputerSciences, Social Informatics and Telecommunications Engineering, M. Gris and G. Yang, Eds., 2012,
[11] Noudjoud Kahya, Nacira Ghoualmi and Pascal Lafourcade, "Formal Analysis of PKM using Scyther Tool", International Conference on Information Technology and e- Services,2012.
[12] Ahmed M. Taha, Amr T. Abdel-Hamid, and SofièneTahar, "Formal Verification of IEEE 802.16 Security Sublayer Using ScytherTool", ESR Groups France, 2009.
[13] Kahya Noudjoud, Debbah Adel and Nacira Ghoualmi, "WiMA Security – A Formal Analysis using Scyther tool", International Conference on Computational Techniques and Artificial Intelligence (ICCTAI'2012) Penang, Malaysia,2012.
[14] The PARI Group, Bordeaux, PARI/GP, version 2.4.3. Available at: pari.math.u-bordeaux.fr,2008.