International Journal of Computer & Information Technologies (IJOCIT)
www.ijocit.org & www.ijocit.ir
ISSN = 2345-3877

Dynamic Group Digital Signature Based on Fractal Sets According to Text Content

HamidReza Biabani Najaf-Abad[1], MohammadAli Jebreil Jamali[2], Shahin Akbarpour[3]

Islamic Azad University of Shabestar[1,2,3]
Iran[1,2,3]
irhamidreza@gmail.com[1], m_jamali@itrc.ac.ir[2], akbarpour_shahin@yahoo.com[3]

Abstract: - Group digital signature lets each member of a group sign a message on behalf of the entire group, in a way that the identity of signers is not revealed. In this paper, a new method for group digital signature based on Mandelbrot and Julia fractal sets according to Text Content is presented. In this method, the concept of documents text content is used in order to categorize the members whose identity is confirmed for each text. This scheme may be used for signing sensitive documents. Texts and documents can be categorized into classified, secret, and top-secret according to their sensitivity. The group manager selects qualified authorized people to sign documents according to text type. In this method, group members are placed on the leaves of a binary tree structure and sign texts while interacting with each other. The presented method resists coalition attacks and enjoys larger key space compared to RSA and DSA algorithms. The public key size in the presented method is independent of the number of group members. Moreover, group departure protocol is also incorporated in this method and the method is thus completely dynamic. The members' position in the tree structure is vague in this scheme. This helps to increase security. The fractal dynamic group digital signature presented can be an appropriate substitute for current group digital signature schemes.

Keywords: Group digital signature, Binary tree structure, Mandelbrot and Julia sets, Classified documents.

1. Introduction

Digital signatures are used nowadays in many countries as powerful tools to confirm contracts and verify identities. It's scheme does the work of manual signature while guaranteeing more security. Furthermore, digital signatures provide secure exchange distribution for public keys. Hence, they are used as the basis of all public key cryptography. In a group digital signature scheme, members of a group digitally sign a document on behalf of the entire group. In addition, signatures may be verified only via a group public key. Once a document is signed, no other person except for the one who has set up the group can specify which group member has signed the document. Companies use group signatures to give credit to price lists or digital contracts. Customers only need to know a public key of the company to confirm the signature. Companies can keep their internal structure hidden and are also able to specify which employee has signed the document. Group signatures should be designed in a way that no group member is able to forge other members' signatures. The application of group signatures is in cases where the receiver needs to know that the signing has been done by a group and the group members are not important.

The first idea of a digital signature was presented by Whitfield Diffie and Martin Hellman in 1976 [1]. Afterward, Rivest, Shamir, and Adelman invented the first digital signature algorithm called RSA [2]. Later on, other digital signature algorithms such as ElGamal [3] and undeniable signature [4] were invented. Group digital signatures were first presented by Chaum and Van Heyst [5]. Four schemes are presented in this paper in three of which the group manager is required in order to establish connections among members and find out which member has signed a particular document. In two out of the four presented schemes in this paper, it is impossible to add a new member to the scheme. Chen and Pedersen [6] proposed two group signature schemes. Unfortunately, their schemes

had this problem that the group manager could sign messages, in a way that it seemed as if other members have signed. Another paper on group signatures is written by Camenisch [7]. A group signature was presented in this paper which was more efficient than previous works. In this scheme, the group manager is unable to sign the text on behalf of other members and besides, once the group is initially founded, new members may be added later. One of the most important group signature schemes is that of Camenisch and Stadler[8]. All security measures are observed in this scheme and the size of group public key is constant. After that, Ateniese and Tsudik [9, 10] conducted some research on group digital signature. These schemes perform poorly against coalition attack. The first practical group signature was presented by Ateniese and Tsudik [10] in 1999 based on RSA strong assumption. This scheme was then bettered and proved in a formal model [11]. The weakness of this signature was that members' membership could not be cancelled. ACJT group signature scheme [12] was the first practical scheme to resist coalition attack and is provable in terms of security. Nevertheless, the possibility of members' membership cancelation without great computational cost being added was one of the problems of this scheme. Numerous experts presented solutions such as using membership cancelation lists by considering time periods. One of the proper solutions was presented by Camenisch and Lysyanskaya [13] which introduced dynamic saving scheme. Group signature of Camenisch and Groth [14] is based on that of ACJT in which membership cancelation is also incorporated. Abdolamer Khalaf Hussain [15] presented a new group digital signature based on text content in 2012. In this scheme, which is based on RSA algorithm, signers are selected according to text content type. This method uses other stages for signing sensitive documents according to their classification. In 2013, Min-Shiang Hwang [16] presented a new group signature based on RSA algorithm. In comparison with [17], this scheme

takes less computational time and is more efficient. In 2007, Mohammad Alia et al. presented group key agreement protocol [18] and afterward, they presented digital signature scheme based on Mandelbrot and Julia fractal sets for the first time [19].

This paper presents a new method for dynamic group digital signature based on fractal sets according to text content. The principles are similar to those of the method presented according to the strong connection between Mandelbrot and Julia fractal sets [19] by use of their especial functions Mandelfn and Juliafn [20]. Fractals are complex numbers comprised of two real and imaginary parts. A complex number can be shown as a point in the complex number system. If a complex number is denoted by Z = (a + bi), *a* is the real part of the number shown on the horizontal axis and *b* is the imaginary part shown on the vertical axis. The unit imaginary number is $i=\sqrt{-1}$ [21]. In this paper, Mandelbrot and Julia sets were used, because they were easy to produce and had all of the characteristics of fractals. Merely by changing iteration formula in this program, new keys for signing may be created and the protocol may be improved in terms of security.

✓ **Mandelbrot and Julia Fractal Sets**

Mandelbrot fractal [20] is from a second order function created from recursive relationships in the complex number coordinate system. Figure 1 shows one type of Mandelbrot fractal.

In equation $Z_n = Z_{n-1}^2 + c; Z_0 = 0, c \in C$ the coordinates of Z_n point is substituted in the formula, the square is calculated, the result is added to the constant value *C* which is previously selected and the new point Z_{n+1} is obtained. The square of this point is again calculated, added to *C* and this iteration takes place several times and expansion of *Z* is observed. For example, if we choose *C=-1.0+0.5i*, it is observed that *Z* tends to infinity in few iterations. We typically expect that the square of a number when added to another number grows and tends to infinity. Now, if we choose *C=0.25-0.25i*, we find out that *Z* is in the

vicinity of the initial value. In fact, it converges to a certain point. We saw that *Z* demonstrates different behaviors (convergence as opposed to tending to infinity) depending on selecting the initial point. With this characteristic, all of the points of the complex number plane are substituted in the iteration formula in this fashion: if convergence occurs after a number of iterations, that point belongs to Mandelbrot fractal; otherwise, if it divergence occurs, that point is overlooked.

Figure 1: Mandelbrot Fractal Image

Similar to Mandelbrot fractal set, Julia fractal set (Figure 2) is also a set of points in the complex plane which is defined by an iterative second order relationship (Julia equation $Z_n = Z_{n-1}^2 + c; Z_0 = c, c \in C$).

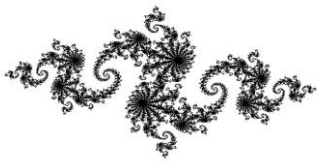

Figure 2: Mandelbrot Fractal Image

The difference between Mandelbrot and Julia sets lies in the fact that the starting point in Mandelbrot set relation is zero, whereas it has a non-zero value in Julia set. If the starting point of Julia set, c, is chosen to be within Mandelbrot set, Julia set will be cohesive and unified. If c is chosen to be outside Mandelbrot set, Julia set will not be cohesive.

2. Dynamic Group Digital Signature based on Mandelbrot & Julia Sets according to Text Content

The proposed protocol is introduced in this section. A binary tree structure is used in this protocol. A binary tree is a tree data structure in which each node has at most two child nodes, often called right and left children. Each user is logically connected to a leaf node from the binary tree. In order to specify each tree node uniquely, the label $<l,v>$ is used where $l \in \{0, \ldots, h\}$ is tree level and h is the height of tree T, and $0 \leq v \leq 2l-1$ is the position of the node at this level. Note that the tree height has a linear

relationship with the number of users in binary linear trees, while h is logarithmic in binary balanced trees. In this structure, members form a group. This group has a group manager for managing the group behavior. People are placed on leaves in the tree structure and are merely in connection with their sibling. Each person has two private keys which is a part of the entire group's private key. The entire group also has a public key. Documents are initially classified according to content and text type. Assume $cx=\{cx_1, cx_2, \ldots, cx_n\}$ is the set of texts and $u=\{cx_iu_1, cx_iu_2, \ldots, cx_iu_n\}$ is the set of authorized members for signing cxi class texts. Table 1 shows text types and the respective information.

This scheme can be used for signing sensitive documents. Texts and documents may be categorized into classified, secret, and top-secret documents according to their sensitivity. The group manager selects qualified authorized people for signing documents according to text type. The group manager specifies text types at first and for each text type; authorized signers are specified and placed in Table 1. For signing a text, the group manager initially specifies text sensitivity degree and type and then, constructs the binary tree for signing by having authorized people to sign the text according to Table 1.

Table 1: Context Information Sample Table

Context type	*Users*	*Public Keys*
cx_1	cx_1u_1 cx_1u_2 cx_1u_3	ZM_1 ZM_2 ZM_3
cx_2	cx_1u_1 cx_1u_2	ZM_1 ZM_2
.	.	
.	.	
cx_n	cx_nu_1 cx_nu_2 . cx_nu_m	ZM_1 ZM_2 . ZM_m

Once members are selected, group digital signature scheme, introduced in what follows, is carried out. This scheme has the following 6 procedures: preparation, joining the group, leaving the group, signing the message, signature confirmation, and reopening the signature. Further, these procedures are described.

2.1. Preparation

This procedure includes the following stages:

2.1.1. Generating Parameters:

In this protocol, $(n_i, e_i, c, x, ZM_i, ZJ_i, Mandelfn, Juliafn)$

parameters exist and each member is denoted by u_i. n_i and e_i are the private keys of the person u_i where $n_i \in N$, $e_i \in C$. x and c are public values joint among all members where $x \in N$, $c \in C$. The value of x is used to reduce final calculations. It may be set to zero. The *Mandelfn* function is the Mandelbrot function with the equation $z_{n-1} = z_n \times e_i \times c^2$ where $z_0=c$. The Juliafn function is the Julia function with the equation $z_{n+1} = z_n \times e_i \times c$ in which the z_0 value equals the value generated by the *Mandelfn* function. ZM_i is the public key of user i obtained with the help of *Mandelfn* relation. ZJ_i is also the output of *Juliafn* relation for user u_i. Parameters c and x are generated by the group manager and sent to all users.

2.1.2. Generating Users' Private and Public Keys:

User u_i initially chooses n_i and e_i values as its private key, and e_i should belong to Mandelbrot set. Then, using the public value c, which is a complex number in Mandelbrot set, the public key of ZM_i is generated.

2.2. Join

As mentioned before, members are only placed on leaf nodes in this protocol and constitute the values of internal nodes while interacting with each other. When user u_{n+1} intends to join the group $\{u_1,u_2 ,..., u_n\}$, it sends the request for joining the group to the group manager. Since people who intend to join the group have been previously confirmed and are in fact a member of the system and their identity is corroborated before, the group manager responds to this request and at first, enters the new member in the text type row of Table 1 and then specifies a proper location for the new member to be added to the tree structure. As pointed out before, each member in this protocol is only in connection with its sibling (a member which is introduced as the member's brother in the tree structure and they have a joint parent). The group manager introduces the new member's location together with its sibling and starts to manage the new tree. The new member's entry place is the rightmost node with greatest height, in a way

that if possible, addition of a new member to the group should not increase the height of the tree. Two cases exist for the proposed protocol. The cases are explained in what follows.

2.2.1. Tree Height Remaining Constant

This case happens when the binary tree is not complete and the rightmost node is not at $\lceil \log n \rceil$ height (Figure 3). In this case, the new member is added to the group and node *x*, the entry place of the new member, is divided to two parts. The member which was previously in location *x* turns into the child to the left of *x*, (*left(x)*), and the new member turns into the child to the right of *x*, (*right(x)*). For instance, the location of the member u8 in Figure 3 is shown with a dotted line. Since nodes have either two children or no children in this structure, when u8 enters, u7 node turns into a internal node. u7 turns into the internal node's left child and the new member, u_8, becomes the internal node's right child.

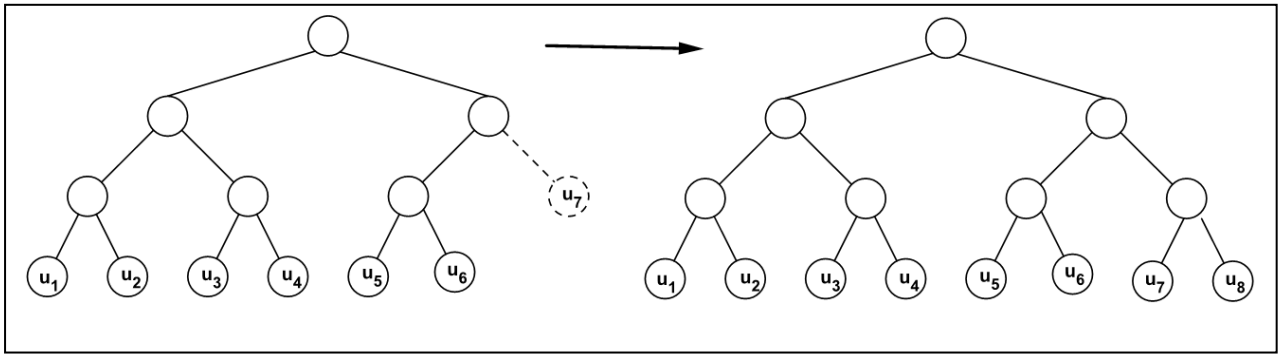

Figure 3: Joining the Group with the Tree Height Remaining Constant

2.2.2. Increase in Tree Height

This case happens when the binary tree is complete and entirely full. In this case, when a new member is added, one unit is added to the tree height. The group manager specifies the entry place of the new member in the group and introduces its sibling. Figure 4 depicts an example of new member addition with increase in tree height.

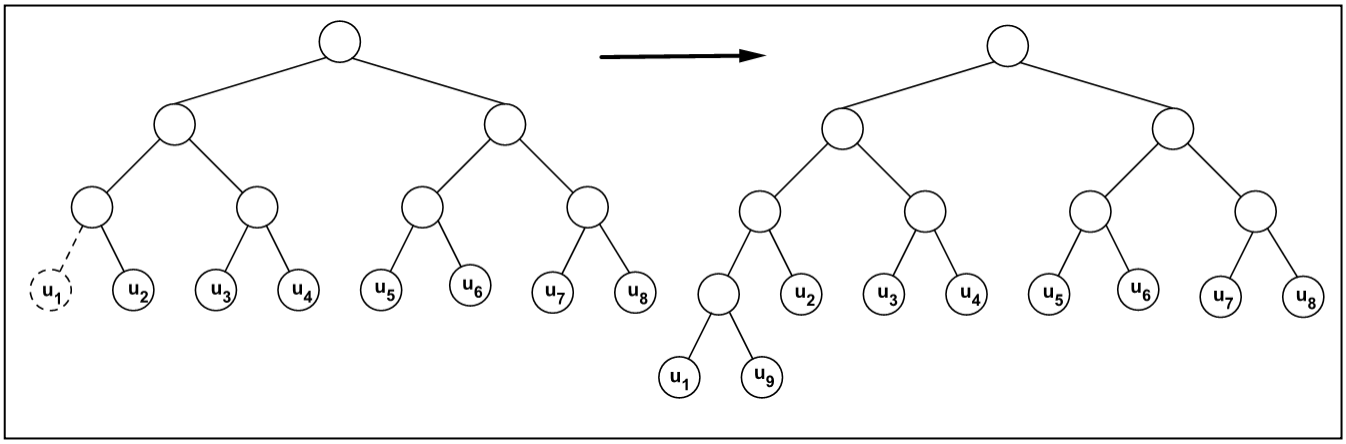

Figure 4: Joining the Group with Increase in Tree Height

Once the location of the new member is specified, the sub tree where the new member has entered should be updated. Afterward, the new member generates signature with the help of its sibling and they update their father node. With the father internal node being updated, the father node and its sibling internal node perform the signature algorithm and they update their father internal node. This process continues until the root node, in which the main signature lies, is updated.

2.3. Leave

When the group manager decides to eliminate a member for some reason or the member itself is not willing to be in the group, the respective member can leave the group with the help of this protocol and without ruining the whole group and its reconstruction. In binary tree structure, elimination of a member takes place only in the last node. In this protocol, at first, member uj, which intends to leave the group, sends a request for the group manager. The group manager replaces the location of this member in the binary structure with the last member (the rightmost node) and then eliminates the last node. With the last node being eliminated, its father node loses one of its children. Since in binary tree each node has either no children (leaves) or precisely two children, when the last node is eliminated, the sibling of the eliminated node replaces its father. Once the respective member is eliminated from

the tree structure, it is also eliminated from the text type row of Table 1. Elimination of a node in this structure gives rise to one of the following cases:

2.3.1. Tree Height Remaining Constant

This case happens when the rightmost node (last node) is at the height $\lceil \log n \rceil$. In this case, the tree height remains constant once the group departure protocol is applied. In Figure 5, member u_3 intends to leave the group and at first, it replaces itself with the rightmost node, i.e. u_4, so that the binary tree structure would not change. Then, the last node, u_3, is eliminated.

2.3.2. Decrease in Tree Height

This case happens when the last node is at the height $\lceil \log n - 1 \rceil + 1$. In this case, with the elimination of each member, the tree height decreases as much as one unit. Figure 6 shows a sample of node elimination with decrease in tree height. In this figure, user u_3 intends to leave the group. At first, the location of this node is replaced with that of the last node and then, node u_3 is eliminated. With this action, node u_1 replaces its father and tree height decreases as much as one unit.

When a member leaves the group, the tree structure changes. Once the group departure protocol is applied, the sub-tree from which the member has departed should be updated.

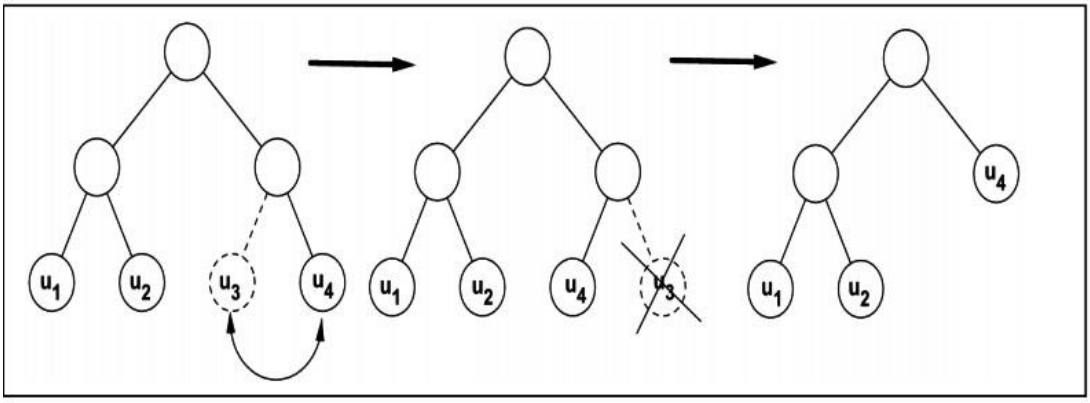

Figure 5: Leave Group With The Tree Height Remaining Constant

In the worst case where the departing member is in one sub-tree (e.g. left side sub-tree) and the last node is in another sub-tree (e.g. right side sub-tree), all of the internal nodes should be updated. But the overall structure of the tree is preserved also in this case and there is no need for its reconstruction, and there will be no cost spent on constructing binary tree.

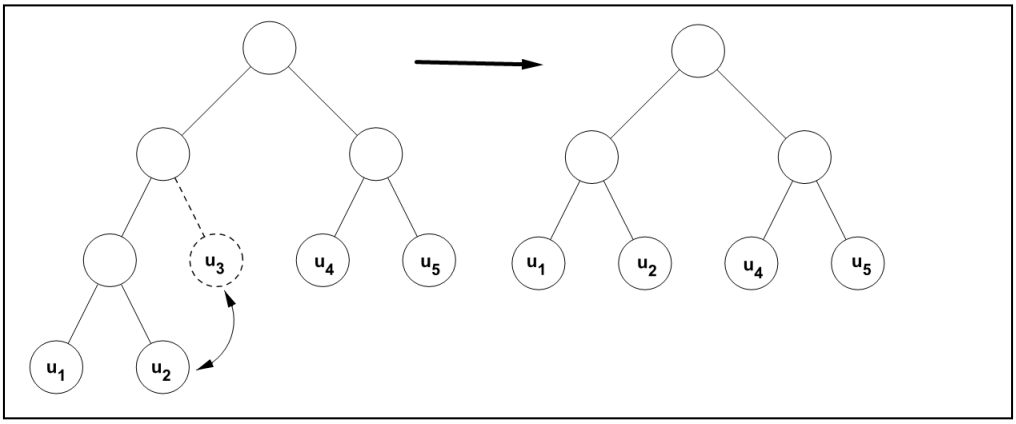

Figure 6: Leave Group with Decrease in Tree Height

2.4. Sign

The proposed protocol is based on symmetrical balanced binary tree structure. In this tree structure, tree leaves represent users. $T<l,v>$ means sub-tree T whose root is placed in node $T<l, v>$. Left and right side children of node $<l, v>$ are denoted by $<l+1, 2v>$ and $<l+1, 2v+1>$ indices, respectively. Signature calculation in node $<l, v>$ requires calculating Mandelbrot and Julia functions for both children. Signature $S<0,0>$ in the root node is the group joint signature calculated by all members. Each group member in the leaf node $<l, v>$, can calculate the group signature $S<0,0>$ using all public values and its secret key in interaction with its sibling node. In other words, signature $S<0,0>$ in Figure 7 may result from signatures obtained in internal nodes $S<1,0>$ and $S<1,1>$. These nodes are obtained by calculating the signature of their sub-tree leaves. In Figure 7, members are placed on leaves and after the preparation stage, ZM_i values are calculated and

it is used for calculating the father node value. In what follows, the protocol execution process of the group digital signature is presented.

Step 1: Each user $u<l, v>$ where $0 \leq v \leq 2l-1$, chooses $n_i \in N$ and $e_i \in C$ randomly. There are also public keys such as $c \in C$ and $x \in N$ about which all users are informed. Each user substitutes its public and private keys into Mandelbrot function with the equation $z_{n-1} = z_n \times e_i \times c^2$, where $z_0=0$, and the public key, ZM_i, is obtained.

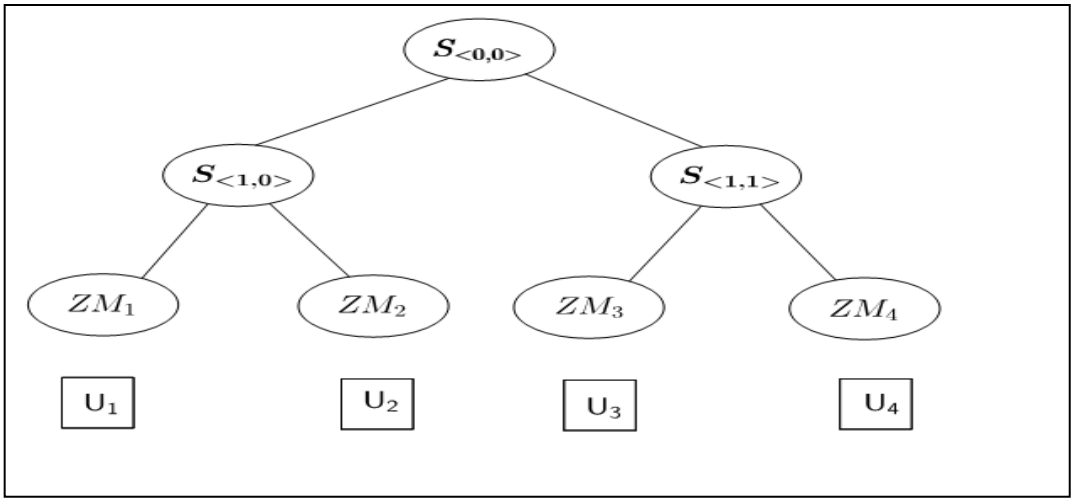

Figure 7: Sign Protocol

Step 2: Each user sends its public key calculated in step 1, ZM_i, for its sibling. Then, each member substitutes its sibling's public key into Julia function with the equation $ZJ_i = Zn = Z_{n-1} \times e_i \times c$, where $Z_0=ZM_j$, and ZMj is the public key of the user's sibling, and the ZJ_i value is calculated.

Step 3: In this step, the signatures of both siblings are calculated and placed in their father node. In order to achieve node $S<l,v>$ signature, the right side child calculates $S<l+1,2v+1>$ value via Equation (1).

$$S_{<l+1,2v+1>} = ZJ_{<l+1,2v+1>} \times e_{<l+1,2v+1>} \times c^{n_{<l+1,2v+1>}-x} \times m \quad (1)$$

The left side child also calculates the S<l+1,2v> value simultaneously using Equation (2).

$$S_{<l+1,2v>} = ZJ_{<l+1,2v>} \times e_{<l+1,2v>} \times c^{n_{<l+1,2v>}-x} \times m \quad (2)$$

If equality $S<l+1,2v> = S<l+1,2v+1>$ holds, both children have calculated valid signatures.

Now, the $S<l+1,2v+1>$ value is substituted as the private key, $e<l+1,2v+1>$, of node $S<l,v>$. Then, a random number $n<l,v> \in N$ is chosen as the second private key of this node. In other words, for private keys of $S<l,v>$ node, we have: $e<l,v> = S<l+1,2v+1>$, and $n<l,v>$ is a random number.

Step 4: Steps 1 to 3 are repeated for other internal nodes so that the final signature would be placed in the tree root.

2.5. Verify

In order to confirm signature *S* on message m, the confirmer needs to know the group public key. Once the group signed message *M* with the Equation (3), where ZJ_v is the confirmer's public key and ng and eg are group private keys, it sends message *M* together with signature *S* for the confirmer. The confirmer can confirm the signature using Equation (4) and consider the message to be valid.

$$S = c^{n_g - x} \times e_g \times m \times ZJ_v \quad (3)$$

$$V = c^{n_v - x} \times e_v \times m \times ZJ_g \quad (4)$$

In Equation (4), n_v and e_v are confirmer's private keys and ZJ_g is also the group public key. Signature *S* is valid if we have $S=V$.

2.6. Open

Having signature *S* on message *m*, the group manager can recognize signers. Since the group manager is a trustworthy person and it is difficult to gain access to his information, the group manager saves signers for each signature so that if needed, signers can be recognized. Names of signers as well as their location are saved on a file in the tree structure via one way hash function algorithm, accessible only to the group manager. In the signature reopening process, it may only be examined whether the respective person took part in signing or not. The identity of other signers may not be understood. If unauthorized people access this file, they are unable to recognize the people of the group, because by having saved values in one way hash function algorithms, signers' names may not be accessed. The security of this protocol depends on that of one way hash function algorithms.

2. Analysis and security of the proposed protocol

A part of the proposed protocol's security similar to two-person digital signature based on Mandelbrot and Julia sets [19] is because of fractal functions' disorderly characteristics. Because of the number of iterations, n_i, and also variety in private key e_i for members as well as disorderly characteristics of functions, it is very difficult and complicated to design an attack for this scheme, because a slight change in one of the values leads to a major change in result. If 128 bits are used to show e_i values, there are 2^{128} possible cases altogether which is a very huge number, and the probability of finding the correct value would be 1/128. This prevents private values of being attacked.

Using binary tree structure in this protocol also increases security. When a member changes location, signature changes completely. In other words, one of the other important criteria in this group signature is the placement order and location of members in the tree structure. The group public key in this protocol is different for each signature and each signer has two private keys and one public key in this method. Signers gradually make private keys and the public key of the group. In other words, private and public keys of each signer are a part of group private and public keys. Signature generation in this protocol is made more secure and stronger by considering these parameters. Besides, the security of this signature increases owing to the use of multiple private keys instead of only one.

This signature is made by members whose identities have been verified and fulfills the security needs required by group digital signature. Members are selected according to text type in this signature. Signatures made by this group and by use of signature algorithm are confirmable by the confirmation algorithm. The generated signature cannot be forged and only group members can generate a valid signature, confirmable with the group public key. Group members are from one organization and their identities have been previously verified. Moreover, each member is added to the group

with the permission of the group manager. This also guarantees that the signature is not forgeable so that unauthorized people would not be able to forge this signature. In this signature, only the group manager may specify which member has created the signature. Even signers are not aware of the identity of other signers. The group manager can always specify the identity of group members who cooperated in a signature, because the group manager saves signers and their location in the binary tree structure using one way hash functions for each signature. Furthermore, this signature also resists coalition, as all activities in this signature are controlled and managed by the group manager. It is not possible for signers to create a valid signature through sharing their private keys approved by the confirmation process, while signers are not known by running signature reopening process. The reason lies in the fact that each person's location in the tree structure is especially important in addition to people's private keys. This is managed by the group manager. Hence, no subgroup can create a valid signature without interacting and cooperating with the group manager. No group member may sign for another, because nobody knows about other signers' private key and needs the group manager's interaction and confirmation in order to enter the group.

Table 2 is a practical example of group digital signature based on Julia and Mandelbrot fractal sets. This example is simulated in MATLAB R2011a software. In this example, the public value of c is *(-0.2534) + (-0.467)i* and the value of *x* is set to *0*. The group manager selects members according to text type and the selected members sign the text by performing the proposed protocol. Group members initially choose their private keys (Table 2, row 2). Then, they generate their respective public keys using *Mandelfn* function (Table 2, row 3). The fourth row of Table 2 shows that siblings send their public keys for each other. Afterward, user *i* generates its signature using its private key (Table 2, row 5) and then places the signature in

its father node. The father node uses it as its private key and calculates its public key (Table 2, row 6). In row 7 of Table 2, signer u_3 and node *<1,0>* send their public keys for one another. In row 8 of Table 2, node *<1,0>* calculates its signature and places it in the root node *<0,0>*.The signature calculated in the root node is used as group private key and also the random value *n<0,0>* is chosen to be the group private key. The group public key is generated in row 9 of Table 2. The confirmer's (receiver's) public key and the group public key in row 10 of Table 2 are sent for one another. Then, the group signature, *S(G)*, is generated using group private keys and sent for the confirmer along with message m (Table 2, row 11). The confirmer calculates the *S(V)* value and if *S(V)=S(G)*, the receiver confirms the group signature *S(G)*.

Table 2

No.	Description	U1	U2	U3	Verifier(Receiver)
1	c = -0.2534 -0.467 ; m=0.0123 + 0.0321; x=0				
2	Generate the private keys	n(1)=1 e(1) = -0.1380792 - 0.180792	n(2) = 2 e(2) = 0.13407807929 + 0.134043	n(3)= 3 e(3) = 0.1935123213 + 0.8217323	n(V) = 1 e(V) = ev=0.024535 + 0.015643
3	Generate the public keys by using Mandelfn	PK(1) = -0.018496503904476 - 0.028673399178070	PK(2) = -0.001206487772547 - 0.000927812085165	PK(3) = -0.003688200973141 + 0.006174175650560	PK(V) = 0.010685300745718 + 0.000849504533932
4	Exchange the public keys between siblings	PK(2) = -0.001206487772547 - 0.000927812085165	PK(1) = -0.018496503904476 - 0.028673399178070		
5	The user(i) finds his signature by Juliafn	S(1) = 7.132023601298862e-07 - 3.283429023575601e-06	S(2) = 7.132023601298862e-07 - 3.283429023575601e-06		
6	complete the parent nodes (<1,0>)	S<1,0> = e<1,0> = 7.132023601298862e-07 - 3.283429023575601e-06 n<1,0>= 1 PK<1,0> = 1.456687221000000e-07 - 4.824605110000000e-07			
7	exchange the public keys between siblings	PK(3) = -0.003688200973141 + 0.006174175650560		PK<1,0> = 1.456687221000000e-07 - 4.824605110000000e-07	
8	complete theroot node (<0,0>)	S<0,0> = e<0,0> = -6.623025786738656e-09 + 1.634401889658973e-09 n<0,0> = 2			
9	Genarate the public key of group	PK(G)=-3.371342516902227e-19 + 1.941397338798553e-18			
10	Exchange the group public key and the public key of verifier	PK(V) = 0.010685300745718 + 0.000849504533932			PK(G)=-3.371342516902227e-19 + 1.941397338798553e-18
11	The group manager sends S and m to the receiver	S(G) = 5.100558545869873e-22 - 1.267770805480264e-21			
12	Receiver must calculates S(V), if S(V)=S(G) Receiver verifies the signed message				S(V) = 5.100558545869873e-22 - 1.267770805480264e-21

3. Conclusion

In this paper, a new dynamic group digital signature protocol based on Julia and Mandelbrot fractal sets according to text content was presented. this method considers the context type of each document in order to classify them according to their sensitivity. So, this sensitive

document is signed by a group of users designated for that context type. In the protocol introduced, complete binary tree structure is used. In this structure, each leaf represents a user. In this protocol, each user signs the message with the collaboration of its sibling in the tree structure and places the signed message in its father node. Internal nodes use these signatures as their private key and sign the message with the cooperation of their sibling in the tree structure and place it in their father node. This process continues until the signature reaches the root node. This method pays special attention to the text type of each document and signers are selected by the group managers according to text content. Compared with RSA and DSA algorithms, the proposed protocol can generate more keys in a fewer number of bits. In addition, this group digital signature is dynamic and it also includes a group departure protocol. This protocol provides higher security by employing the characteristics of chaotic systems.

References

[1] W. Diffie and M.Hellman, "New directions in cryptography," *Information Theory, IEEE Transactions*, vol. 22, no. 6, 1976, 644–654.

[2] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, 1978, 120-126.

[3] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions*, vol. 31, no. 4, 469 – 472, 1985.

[4] B. Pfitzmann and M. Waidner, "Fail-stop signatures and their application," 1991.

[5] D. Chaum and E. Van Heyst, "Group signatures," in *Advances in Cryptology—EUROCRYPT'91*, pp. 257–265. Springer, 1991.

[6] L. Chen and T. P. Pedersen, "New group signature schemes," in *Advances in Cryptology—EUROCRYPT'94*, pp. 171–181. Springer, 1995.

[7] J. Camenisch, "Efficient and generalized group signatures," in *Advances in Cryptology—EUROCRYPT'97*, pp. 465–479. Springer, 1997.

[8] J. Camenisch and M. Stadler, "Efficient group signature schemes for large groups," in *Advances in Cryptology—EUROCRYPT'97*, pp. 410–424. Springer, 1997.

[9] G. Ateniese and G. Tsudik, "A coalition-resistant group signature scheme," Technical Report, in Submission, 1998.

[10] G. Ateniese and G. Tsudik, "Group signatures á la carte," in *Proceedings of the ten-th annual ACM-SIAM symposium on Discrete algorithms*, pp. 848–849, 1999.

[11] A. Kiayias and M. Yung, "Group signatures: Provable security, efficient constructions and anonymity from trapdoor-holders," *IACR*

Cryptology ePrint Archive 2004, pp. 76, 2004.

[12] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik, "A practical and provably secure coalition-resistant group signature scheme," in Advances in Cryptology—EUROCRYPT'2000, pp. 255–270. Springer, 2000.

[13] J. Camenisch and A. Lysyanskaya, "A signature scheme with efficient protocols," in Security in communication networks, pp. 268–289. Springer, 2003.

[14] J. Camenisch and J. Groth, "Group signatures: Better efficiency and new theoretical aspects," in Security in Communication Networks, pp. 120–133. Springer, 2005.

[15] A. Hussein, "Generating a new group digital signatures," Journal of Emerging Trends in Computing and Information Sciences, vol. 3, no. 6, 2012.

[16] C. C. Yang, T. Y. Chang, and M. S. Hwang, "A new group signature scheme based on rsa assumption," Information Technology And Control, vol. 42, no. 1, 2013, 61–66.

[17] C. C. Lee, T. Y. Chang, and M. S. Hwang, "A new group signature scheme based on the discrete logarithm," Journal of Information Assurance and Security, vol. 5, no. 1, pp. 54–57, 2010.

[18] M. A. Alia and A. Samsudin, "New key exchange protocol based on mandelbrot and julia fractal sets," International Journal of Computer Science and Network Security (IJCSNS), vol.7, no. 2, pp. 302–307, 2007.

[19] M. A. Alia and A. Samsudin, "A new digital signature scheme based on mandelbrot and julia fractal sets," American Journal of Applied Sciences, vol. 4, no. 1, pp. 850–858, 2007.

[20] B. B. Mandelbrot, "The Fractal Geometry of Nature," Macmillan, 1983.

[21] E. Patrzalek, "Fractals: Useful Beauty General Introduction to Fractal Geometry," 2006.

