A BODY-CENTERED CUBIC METHOD FOR KEY AGREEMENT IN DYNAMIC MOBILE AD HOC NETWORKS

Ioannis G. Askoxylakis
Foundation for Research & Technology – Hellas, Institute of Computer Science (FORTH-ICS)
asko@ics.forth.gr

Damien Sauveron,
XLIM
University of Limoges
Damien.Sauveron@xlim.fr

Konstantinos Markantonakis,
Royal Holloway University of London
Information Security Group, UK
K.Markantonakis@rhul.ac.uk

Theodore Tryfonas
Faculty of Engineering
University of Bristol
ttryfona@glam.ac.uk

Apostolos Traganitis
Foundation for Research & Technology – Hellas, Institute of Computer Science (FORTH-ICS)
tragani@ics.forth.gr

Abstract

Mobile ad hoc networking is an operating mode for rapid mobile host interconnection, where nodes rely on each other, in order to maintain network connectivity and functionality. Security is one of the main issues for mobile ad hoc networks (MANETs) deployment. We introduce a weak to strong authentication mechanism associated with a multiparty contributory key agreement method, designed for dynamic changing topologies, where nodes arrive and depart from a MANET at will. We introduce a new cube algorithm based on the body-centered cubic (BCC) structure. The proposed system employs elliptic curve cryptography, which is more efficient for thin clients where processing power and energy are significant constraints. The algorithm is designed for MANETs with dynamic changing topologies due to continuous flow of incoming and departing nodes.

1. Introduction

1.1 Motivation for group key agreement

Consider a military operation, where the need for establishing a network quickly and securely is crucial. The potential network members seek to establish a wireless network in the most secure and efficient way. The objective is to interconnect all computing and communication devices where they will be able to share all necessary information securely, since nobody can guarantee that the "high tech" enemies will not try to disrupt or intercept the operation efforts.

The technical goal is to make sure that no other entity outside the *group* (*we define all the legitimate members of the established wireless network as group, e.g., soldiers of a military unit*) should be able to gain access within the new network. However, since neither a certification authority nor a secure communication channel exists, the enemy has the ability to eavesdrop and modify exchanged messages transmitted over the air. Additionally, since no central identification authority is present, group member impersonation is easy, jeopardizing the security of the whole system.

Considering all these issues, the main challenge that arises is the setting up of a wireless network where the legitimate members of a group will be able to establish a secure wireless network. Moreover, in the case where a new node arrives at place, desiring to become a member in an already established group, joining, without delaying or even intercepting the existing group, is also challenging. Finally, we consider the case where a group member is captured by the enemy and therefore the group key is compromised.

1.2 Security and MANETs

Security is an important factor in the design of decentralized communications systems, like ad hoc networks. An ad hoc network is a type of network, which is typically composed of equal mobile hosts that we call *nodes*. When the nodes are located within the same radio range, they can communicate directly with each other using wireless links. This direct communication is employed without the presence of a central control. The absence of backbone infrastructure, such as base stations, introduces several problems, such as configuration advertising, discovery, maintenance, as well as ad hoc addressing, self-routing and security. Since no central certification authority exists, trust cannot be provided among the nodes of the network without the existence of initial specific *prior known information*. This special kind of information is necessary in order to build trust between all participating nodes. We define that an ad hoc network is *established* among the existing nodes, if from preexisting, commonly known information, we reach a state where a common *Session Key* is agreed. Securing ad hoc networks can be very challenging, mainly because of

978-0-7695-3329-2/08 $25.00 © 2008 IEEE
DOI 10.1109/SECURWARE.2008.42

IEEE computer society

the dynamic topology and the vulnerability of the wireless links, which allows passive and active attacks.

1.3 From weak to strong authentication

When dealing with system security, we must always keep in mind that security mechanisms cannot create trust. In our example, the members of a military team know and trust one another in advance. If they did not, they would never be able to achieve mutual trust regardless of the authentication mechanism used. Therefore, our goal is to exploit the existing physical mutual trust in order to secure the ad hoc network. A password authentication approach seems to be an efficient scheme that can deliver a proper solution without the need for additional requirements (i.e., smart cards). A first approach could be the use of a sufficiently large and randomly generated data string that can be used as a password. Based on this, all nodes can agree on a password and, by using a trivial authentication protocol, can easily achieve mutual authentication. However in a scenario like this, the underlying security depends on the size and the randomness of the chosen password, which the larger it gets the more difficult it is to be memorized and used. Moreover, since the response time is limited and vital during military operations, the use of large passwords can be proved very inconvenient and problematic. Therefore the use of short, user-friendly passwords is essential. On the other hand, the use of short passwords provides weak authentication since the password selection set is quite limited and thus every authentication procedure is vulnerable to dictionary attacks [1]. To overcome this, we need an authentication protocol that will lead to a reasonable degree of security even if the authentication procedure has been initiated from a small, weak password.

1.4 Security requirements

We can classify security threats into two main categories depending on the origin of the attack. We distinguish attacks as external and internal. The first originate from the external environment and the latter from group authenticated nodes. For instance, consider a group of soldiers operating in a hostile environment, trying to keep their presence and mission totally unknown from the enemy, and the case where a soldier, member of the mission group, is captured by the enemy who is now in a position to attack from within. Another example, less extreme, is an ad hoc network formed in a classroom during a test exam between the laptops or PDAs of the students and the teacher's workstation. According to this scenario, not only we must secure the network from an external intruder but also from a student who exits the classroom in order to retrieve the solutions and then returns. In all these cases, the badly behaving nodes must definitely be expelled from the established network. Before moving into any further details, we need to outline the significant security properties for designing secure MANETs.

Secure authentication. This property defines that only the entities that hold the correct password will eventually become members of the MANET.

Forward authentication. This property defines that even if a malicious partner manages to compromise a network entity in a later phase, he will still be unable to participate in the already existing network.

Contributory key establishment As we have stated, an ad hoc network is established when a session key is agreed among all network nodes. A single entity can easily define a session key, after the authentication procedure is successfully accomplished, and enforce it to the remaining ones. However this solution may become an important security threat, since if this node is compromised the entire MANET security will break down. Therefore the session key should be generated throughout a process where all participating entities contribute (equivalently or hierarchically, i.e., according to the military ranking)

Resilience against attacks. The network must be tolerant against active and passive attacks Even if an attacker inserts, deletes, or modifies the messages exchanged among legitimate nodes, the network must remain intact and the underlying security should not be affected.

The rest of the paper is organized as follows: In Section 2, we start with a review of the previous work concerning two-party and multiparty key agreements and we give a brief introduction on weak to strong authentication and the elliptic curve theory. We describe the state of the art in multiparty key agreement protocols and particularly the d-cube and aggressive d-cube algorithms and examine their properties. In Section 3, we propose a new algorithm for the dynamic case, where the composition of the network topology changes with time, as the network nodes are entering or exiting the network. A discussion concerning the implementation issues and the problems that arise is presented in Section 4. Section 5 concludes this paper by remarking on some issues for future work.

2. State of the art

2.1 Password-based key exchange

Classical cryptographic protocols based on keys chosen by the users are weak to dictionary attacks. Bellowin and Merrit [2] proposed a protocol called *encrypted key exchange (EKE)* where a strong shared key is derived from a weak one. The basic concept of the generic protocol is as follows: there are two parties A and B that share a password P. Both parties use a suitable symmetric cryptosystem, but entity A has also the ability to create a random asymmetric key pair (e_A, d_A). During the first step, A generates a random public key e_A and encrypts it symmetrically using key P in order to produce P(e_A). Then A sends it to B,

$$A : (A_{id}, P(e_A)) \to B. \qquad (1)$$

This message includes A's id in clear text.

Since A and B share the same password P, B decrypts the received message to obtain e_A. Node B generates a random secret key R, and encrypts it in both asymmetric and symmetric cryptosystem using as an encryption key quantity e_A and P, respectively. So B produces $P(e_A(R))$ and sends it to A.

$$A : P(e_A(R)) \to B. \qquad (2)$$

Entity A now decrypts the received message to obtain R, generates a unique challenge $challenge_A$, encrypts it with R to produce $R(challenge_A)$, and sends it back to B,

$$A : R(challenge_A) \to B. \qquad (3)$$

Then B decrypts the message to obtain A's challenge, generates a unique challenge B, and encrypts the two challenges with the secret key R to obtain $R(challenge_A; challenge_B)$. Node B is ready to transmit quantity $R(challenge_A; challenge_B)$ to node A,

$$B : R(challenge_A; challenge_B) \to A. \; (4)$$

When A receives the message, it decrypts it to obtain $challenge_A$ and $challenge_B$, and it compares it with the previous challenge. If there is a match, A encrypts $challenge_B$ with R to obtain $R(challenge_B)$ and sends it to B,

$$A : R(challenge_B) \to B. \qquad (5)$$

If the challenge response protocol has been successfully deployed, then the authentication process is successfully accomplished and both parties proceed, using the symmetric cryptosystem and the quantity R as the session key. However, this protocol has a major drawback: The creation of the common session key R has taken place with unilateral prospective, that is, only by the entity that first initiated the whole procedure. Thus the key agreement scheme is not contributory.

In [3], Asokan and Ginzboorg proposed a contributory version of the above protocol for both two-party and multiparty cases. Their proposal is described as follows:

Two-party case

- $A \to B : A, P(e_A)$
- $B \to A : P(e_A(R, S_B))$
- $A \to B : R(S_A)$
- $A \to B : K(S_A, H(S_A, S_B))$
- $B \to A : K(S_B, H(S_A, S_B))$

where S_A, S_B are the random quantities generated from A, B respectively and K is the session key produced according to the formula $K = F_1(S_A, S_B)$, where F_1 is a one way function and $H()$ is a public hash function.

Multiparty case

- $M_n \to ALL : M_n, P(E)$
- $M_i \to M_n : M_i, P(E(R_i, S_i)), i = 1, ..., n-1$
- $M_n \to M_i : R_i(\{S_j, j = 1, ..n\}), i = 1, .., n-1$
- $M_i \to M_n : M_i, K(S_i, H(S_1, ..., S_n)), forsomei$

where E is the public key of M_n. $S_i, \forall i$ are the random quantities generated from M_i, and K is the session key produced according the formula $K = F_2(S_i), \forall i$. F_2 is a n-input one-way function and $H()$ is a public hash function.

2.2 Password-based Diffie–Hellman key exchange: Two-party key exchange

Diffie–Hellman is the first public key distribution protocol that opened new directions in cryptography [3]. In this important key distribution protocol, two entities A, B, after having agreed on a prime number p and a generator g of the multiplicative group Z_p, can generate a secret session key. In [2], Bellovin and Merritt proposed a password-authenticated key exchange that operates in the following way.

- A picks a random number R_A, calculates $P(g^{R_A}(modp))$, and A sends $A, P(g^{R_A})$ to B; Entity A's id is sent in clear text.
- B picks a random number R_B and calculates $g^{R_B}(modp)$. B uses the shared password P to decrypt $P(g^{R_A} modp)$ and calculates
$$g^{R_B R_A} modp. \quad (6)$$
- The session key K is derived from this value by selecting a certain number of bits. Finally, a random challenge, $challenge_B$, is generated. Then B transmits:
$$P(g^{R_B}(modp)), K(challenge_B).$$
- A uses P to decrypt $P(g^{R_B} modp)$. From this, quantity K is calculated; K is in turn used to decrypt $K(challenge_B)$. A then generates a random challenge, $challenge_A$. A sends
$$K(challenge_A, challenge_B).(7)$$
- B decrypts $K(challenge_A, challenge_B)$, and verifies that $challenge_B$ is correct. B sends
$$K(challenge_A). \quad (8)$$
- A decrypts to obtain $challenge_A$, and verifies that it matches the original message.

2.3 Elliptic curve theory

An essential property for the majority of cryptographic applications is the need for fast and precise arithmetic. Calculations over the set of real numbers are slow and inaccurate due to round-off error [4]. Finite arithmetic groups, such as F_p, F_{2^m}, which have a finite number of points, is used in practice. All practical public-key systems today exploit the properties of arithmetic using large finite groups. Additionally, elliptic curves can provide versions of public-key methods that, in some cases, are faster and use smaller keys, while providing an equivalent level of security. Consequently, the use of ECC can result in faster computations, lower power consumption, as well as memory and bandwidth savings. This is very useful for mobile devices, like the ones used in ad hoc networks, which face limitation in terms of CPU, power, and network connectivity.

An elliptic curve [5] consists of elements (x, y) satisfying the equation:

$$y^2 = x^3 + \alpha x + \beta (mod p) \quad (9)$$

for two numbers α, β. If (x, y) satisfies the above equation then $P = (x, y)$ is a point on the elliptic curve.

The elliptic curve discrete logarithm problem (ECDLP) can be stated as follows.

Fix a prime p and an elliptic curve E. Let xP represent the point P added to itself x times. Suppose Q is a multiple of P, so that $Q = xP$ for some x, then the ECDLP is to determine x given P and Q.

The general conclusion of leading cryptographers is that the ECDLP requires fully exponential time to solve. The security of ECC is dependent on the difficulty of solving the ECDLP.

Mathematicians have given considerable attention to the ECDLP. Like the other types of cryptographic problems, no efficient algorithm is known to solve the ECDLP. The ECDLP seems to be particularly harder to solve. Moderate security can be achieved with the ECC using an elliptic curve defined over Z_p where the prime p is several times shorter than 230 decimal digits.

An elliptic curve cryptosystem implemented over a 160-bit field currently offers roughly the same resistance to attack, as would a 1024-bit RSA [6]. However, there have been weak classes of elliptic curves identified such as super singular elliptic curves [7] and some anomalous elliptic curves [8]. Implementations, such as ECDSA [9], merely check for weaknesses and eliminate any possibility of using these "weak" curves [10].

2.4 Elliptic curve Diffie–Hellman

The original Diffie–Hellman algorithm is based on the multiplicative group modulo p. However the elliptic curve Diffie–Hellman (ECDH) protocol is based on the additive elliptic curve group as desribed below. We assume that two entities A, B have selected the underlying field, $GF(p)$ or $GF(2^k)$, the elliptic curve E with parameters a, b, and the base point P. The order of the base point P is equal to n. Also, we ensure that the selected elliptic curve has a prime order to comply with the appropriate security standards [9].

At the end of the protocol, the communicating parties end up with the same value K, which represents a unique point on the curve. A part of this value can be used as a secret key to a secret-key encryption algorithm. We give a brief description of the protocol.

- Entity A selects an integer, $d_A : d_A \in [2, n-2]$. (10)
- Entity B selects an integer $d_B : d_B \in [2, n-2]$. (11)
- A computes $Q_A = d_A \times P$. The pair Q_A, d_A consists A's public and private key.
- B computes $Q_B = d_B \times P$. The pair Q_B, d_B consists B's public and private key.
- A sends Q_A to B, $A : Q_A \rightarrow B$. (12)
- B sends Q_B to A, $B : Q_B \rightarrow A$. (13)
- A computes $K = d_A \times Q_B = d_A \times d_B \times P$. (14)
- B computes $K = d_B \times Q_A = d_B \times d_A \times P$. (15)

Quantity K is now the commonly shared key between A and B. Moreover, it can also be used as a session key. Quantity n is the order of the base point P.

2.5 d-cube protocol

The authentication protocol described earlier, demonstrated the two-party case. It is obvious that for key establishment procedures in ad hoc networks, where several entities are involved, multiparty authentication protocols should be applied. A lot of research has been done in this direction [11], [12]. Becker and Wille [13] presented a method very efficient in terms of number of authentication rounds. According to this method, also known as the d-cube protocol, all entities planning to participate in a network are initially arranged in a d-dimensional hypercube. Each potential network entity is represented as a vertex in the d-dimensional cube and it is uniquely assigned a d-bit address. The addresses are assigned in a way so that two vertices connected along the $i^t h$ dimension differ only in the $i^t h$ bit. There are 2^d vertices, each of which are connected to other d vertices.

2.6 Aggressive d-cube algorithm

In [15], a modified version of [14] called aggressive d-cube algorithm is presented, where faulty nodes are isolated from the ad hoc network during the early stages of the d-cube algorithm. According to the algorithm, the interaction of faulty-legitimate nodes and the chances a faulty node will enter the network by guessing the password are minimized. Moreover, their protocol protects legitimate nodes from unnecessary energy spending, which may be more important in case of thin clients.

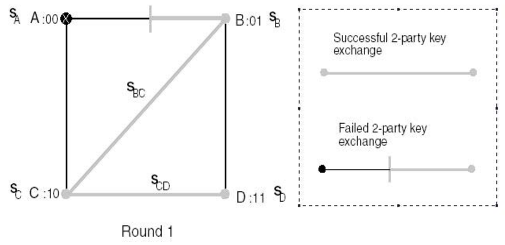
To clearly demonstrate the differences between [15] and [14], we describe the algorithm of [15] through examples in the 2-d and 3-d cases.

2.6.1 Aggressive d-cube algorithm: 2-d example

In this scenario, we considered node A as a faulty malicious partner wishing to compromise the security of our system. Consequently, during round 1 node B will fail to complete the Diffie–Hellman key exchange with node A. However, instead of remaining idle during the particular round, as during the Asokan's case, node B will initiate a new DH key exchange with node C within the same first round. This exchange will be successfully completed, since C is a legitimate partner, resulting in a new key S_{BC} generation. Within the same round, second half of round 1, A will inform C that node B is a faulty member. Meanwhile, node C, during the first half of the first round, has performed a successful key exchange with node D creating shared key S_{CD}. Round 1 will be terminated as soon as the keys S_{BC} and S_{CD} are computed.

During the second round, node B will perform a DH key exchange with node D and the key S_{BCD} will be created. This key will be the common session key among the entities B, C, D. Node C can calculate the common session key for itself, since it has all necessary information needed from the previous round. The key element in this scenario is that during the second round node C will not have to perform any DH key exchange with the faulty node A. Node C has already performed two successful DH key exchanges during the first round, and can compute the common session key S_{BCD} without having to participate in another Diffie–Hellman key exchange. So node A is isolated and moreover loses the ability of performing faulty DH key exchanges with the prospective gain of some valuable exchanged information. That is because its neighbors are contacted within the first round, earlier than in the case of the d-cube algorithm. In the following figures, we present round 1 and round 2 of the aggressive 2-d cube algorithm (Fig. 1).

Aggressive 2-d cube round 1.



Aggressive 2-d cube round 2.

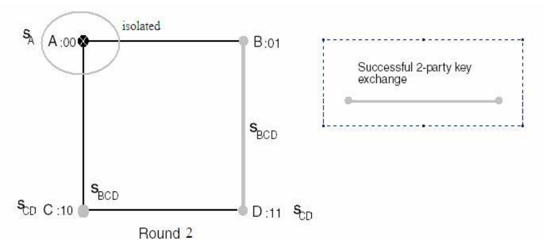


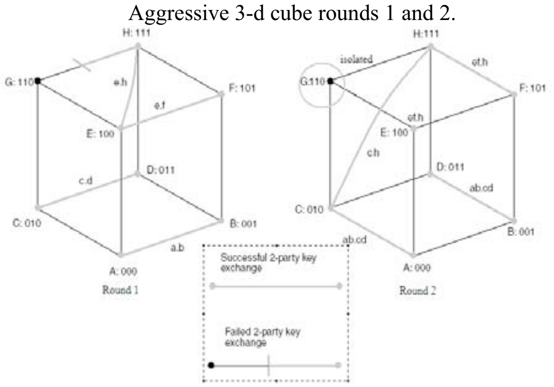
Fig. 1. Aggressive 2-d algorithm

2.6.2 Aggressive d-cube algorithm: 3-d example

In this section we will describe the aggressive 3-d cube example as shown in Fig. 2. In this case we assume that node G is the faulty partner. During the first round, the DH key exchange procedure performed between ($G:110$) and ($H:111$) will fail, since node ($G:110$) is a faulty one. However, instead of remaining idle and waiting for the next round (as in Asokan's case), node ($H:111$) starts a DH key exchange with node ($E:100$). Meanwhile Node ($E:100$) has already performed a successful DH key exchange with ($F:101$), during the first half of the first round, so this key exchange will be the second successful one for this round. Node ($E:100$) having been notified by H that G is a faulty node will remain idle until the third round, instead of having attempted unnecessary DH exchanges with ($G:110$). In the next round (round 2), ($H:111$) performs a DH with node ($F:101$) and a DH with node ($C:010$). Given that ($C:010$) has performed two successful DH with ($D:011$) and ($H:111$) respectively, it will remain idle in the next round. However ($C:010$) has already performed a successful DH with ($A:000$), during round one.

In total node ($C:010$) has performed three successful DH, with three different nodes, which means that ($C:010$) has completed all the appropriate procedures. Thus it will remain idle for the next round, which is the last round in our case. Summarizing the logic of this procedure we would say that the upper bound of the total successful DH procedures for a node participating in an aggressive $d-cube\ algorithm$ is equal to d; in the described example $d = 3$. During the third and final round there will be three more successfully accomplished DH key exchanges: one between ($H:111$) and ($D:011$), one between ($F:101$) and ($B:001$), and one between ($A:000$) and ($F:101$).

Aggressive 3-d cube rounds 1 and 2.



Aggressive 3-d cube round 3.

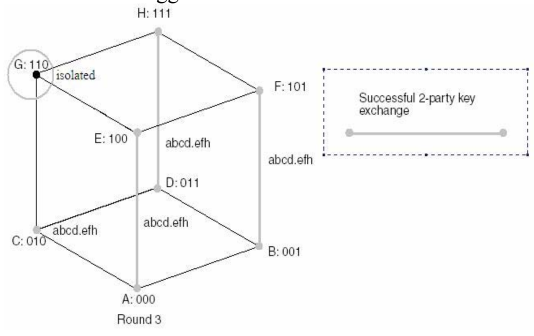


Fig. 2. Aggressive 3-d cube round 3.

Through this example it becomes obvious that using the aggressive 3-d cube algorithm, the faulty partner is being isolated. It only participates in one DH key exchange, i.e., the one performed in round 1 with node ($H:111$), and since then it is excluded from all the subsequent DH key exchanges. Consequently, the faulty node loses the ability to have another change, during the generation process of the common session key, to compromise the security of the system.

3. The proposed system

The dynamic nature of ad hoc networks introduces very challenging security issues. The continuous flow of incoming and departing nodes aggravates the already complicated process of key establishment. We should not forget that when a node *declares* its departure from an active network, it does not lose the ability to "hear" the messages exchanged among the remaining nodes, unless the active session key is refreshed. Therefore, it is crucial only the remaining active nodes contribute in the session key refreshment.

We propose a cryptographic key agreement algorithm that initiates from a tree-arrangement of 3-d cubes; it is based on the aggressive 3-d cube algorithm and employs the *body-centered cubic* (BCC) structure for the dynamic case. For simplicity purposes, in the rest of the paper each bond in 2-d or 3-d space corresponds to a two-party password-based elliptic curve Diffie–Helman key exchange, as described in Subsections 2.2 and 2.4.

3. Initial node arrangement

The proposed system is based on the 3-d aggressive d-cube algorithm [15]. The initial key agreement procedure depends on the number of ad hoc nodes that wish to establish a MANET. We denote the number of nodes as n. In contrast to [15] in the proposed system there is no need for d-dimension hyperspaces. The maximum order is the 3-d space. Nodes of the MANET are always arranged in the 3-d space, except the case that $n \leq 4$ where we can use the 2-d plane. Therefore, when we have a large number of nodes, they must be divided and arranged in 3-d cubes that each contains eight nodes. Each cube selects a leading node that will act as an intermediary between the corresponding cube nodes and the rest of the ad hoc network. The leading nodes constitute a new group; however, they follow the same rules for initial arrangement, i.e., they are arranged in a new 3-d cube. In the case where the number of leading nodes is greater than eight (i.e., the number of all ad hoc nodes is greater than 64), they also need to elect leading nodes in their group that will act as their representatives to the ad hoc network. In such a case, the leading nodes elect higher level leaders in a tree model according to [16].We consider the latter case as an extreme case since from a practical point of view typical ad hoc networks do not exceed 64 nodes. Figure 3 depicts an initial arrangement of a 32-node MANET. Nodes are arranged in four independent cubes and each cube elects a leader (dashed annotation). Node arrangement and addressing can be performed in any way, as far as every simple-cube node has wireless connection with the rest of the seven nodes of the corresponding cube. This requirement must be also fulfilled by the leading nodes among themselves; therefore it is an important criterion for the selection of a leading node within a simple-cube.

3.2 Initial (static) key agreement

The next step, after the initial 3-d arrangement, is to create a common network key. In the proposed system this is done in two steps.

During the first step, the leading nodes perform a 3-d aggressive cube algorithm and they create a global session key. In the second step, every group performs a 3-d aggressive d-cube and establishes a simple-cube session key. During the simple cube key generation, the leading nodes transmit the global session key that they have already established in step 1 to the remaining seven nodes of the group. After the second step, every node has a contributory simple-cube session key K_{cube} for the cube that is part of, and the global session key of the entire

MANET K_{global}. Figure 3 depicts the initial key agreement for a 32-node MANET.

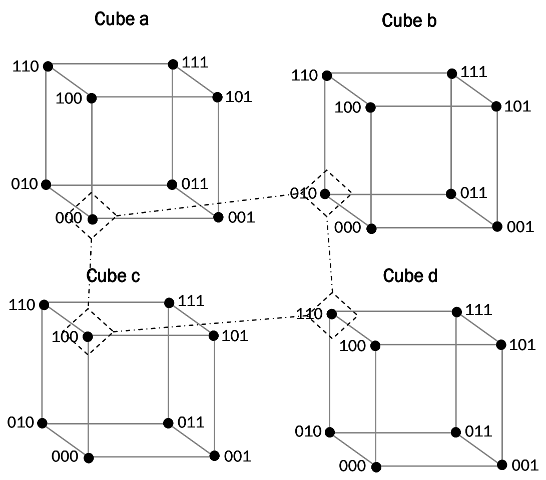


Fig. 3. A 4-cube example.

In the first step nodes (000) of cube a, (010) of cube b, (100) of cube c, and (110) of cube d are elected as leading nodes of the corresponding cubes. Since they are four, they perform a 2-d aggressive algorithm as shown in Fig. 1 and they establish a global session key K_{global}. If we had more than four and equal or less than eight 3-d cubes, their leaders should perform a 3-d aggressive cube algorithm as shown in Fig. 2. In this example, the leading nodes can use the first two digits of their addressees as a 2-d address for the 2-d aggressive algorithm, i.e., (00), (01), (10), and (11). If other nodes are elected as cube leaders due to communication constraints, they should be addressed in a separate way than the one employed in their 3-d cube (second addressing is required).

```
proc do_rnd_subrnd(rnd_nbr,subrnd_nbr)
mask:=00..01 //initialization.
mask:= mask << rnd_number-1 // left shift.
partnet := my_address XOR mask.
new_mask :=  mask >> 1 //right shift mask.
TPDH(partner, new_mask) // two party D-H.
if(result=fail)

/* In case of a failed D-H, the node
is seeking for a new partner,
within the same round
but within a different subround.
The new partner is the next partner
of the faulty node of this round.*/

mask:=mask << sround_number
new_partner:= my_address XOR mask
new_mask := mask >> 1
TPDH(partner,new_mask)
endif
```

Fig. 4. Aggressive 3-d cube algorithm

Once the global session key of the entire MANET K_{global} is established the cubes perform a 3-d aggressive d-cube and they establish the simple-cube session key K_{cube}. The final step is that each cube leader broadcasts the global key encrypted with the simple-cube key to the rest of the cube members. At the end of the protocol, every node has a simple-cube session key K_{cube} for secure communications among nodes of simple-cubes, and a global session key K_{global}, for the entire MANET.

3.3 The body-centered cubic algorithm for the dynamic case

In Sections 3.1 and 3.2, we described the initial arrangement-addressing of nodes and the generation of a global and of simple-cube session keys. These keys are static, since if we want to add new nodes to the network the key generation procedure must be repeated. In this subsection, we describe an efficient method for dynamic key generation every time new nodes arrive to or depart from our network. The proposed dynamic algorithm is based on the body-centered cubic structure and we call it BCC algorithm.

The body-centered cubic (bcc) structure is a cube with an additional node in the center. Figure 5a shows a typical cube while Fig. 5b depicts a body-centered cube. If we consider the grid case, the bcc structure is a set of bcc cubes. The BCC algorithm for dynamic changing topologies is presented through three cases: addition of new nodes to an established MANET, extraction of MANET nodes, and both addition and extraction of nodes.

Case 1: Adding nodes to an established MANET

The BCC algorithm operates in the following way: Assume that a MANET has been established according to Section 3.2. Assume that one simple cube of this MANET is depicted in Fig. 5a. At some point of time, seven new nodes arrive and request to join the MANET. If the number of the new arriving nodes m is a multiple of 8 i.e., $m \bmod(8)=0$ then in groups of 8 they perform aggressive cube algorithms and each group elects a leader that will contact leaders of new groups and leaders from the established MANET in order to create a new global session key. If $m \bmod(8)\neq 0$ then we will have k new groups of 8 nodes where k is the integer part of m/8 and l the number of the remaining nodes where $0< l=m \bmod(8)<8$ while the k groups of 8 nodes will perform new aggressive cube algorithms, the remaining node will attach to an existing cube of the MANET in the following way:
the first four new nodes are assigned addresses that correspond to the center of the existing cube the centers of the right, upper, and front cubes as shown in Fig. 5c while the last three are assigned addresses that correspond to the centers of the left, back, and down cubes as shown in Fig. 5e. We should clarify that the six neighboring cubes do not exist as MANET cubes; they are used as geometrical objects for demonstration purposes of the BCC algorithm.

The first four new nodes (the body-centered cubic node and three central nodes of neighbor cubes) create a new cube with four nodes of the preexisting MANET cube as shown in Fig. 5d and they perform a new aggressive cube algorithm. The latter 3 new nodes together with the body-centered cubic node (the node assigned to center of the existing cube of the MANET that has already participated in the previous new aggressive cube) and the remaining four nodes of the preexisting cube of the MANET perform a second new aggressive cube algorithm as shown in Figs. 5e and 5f.

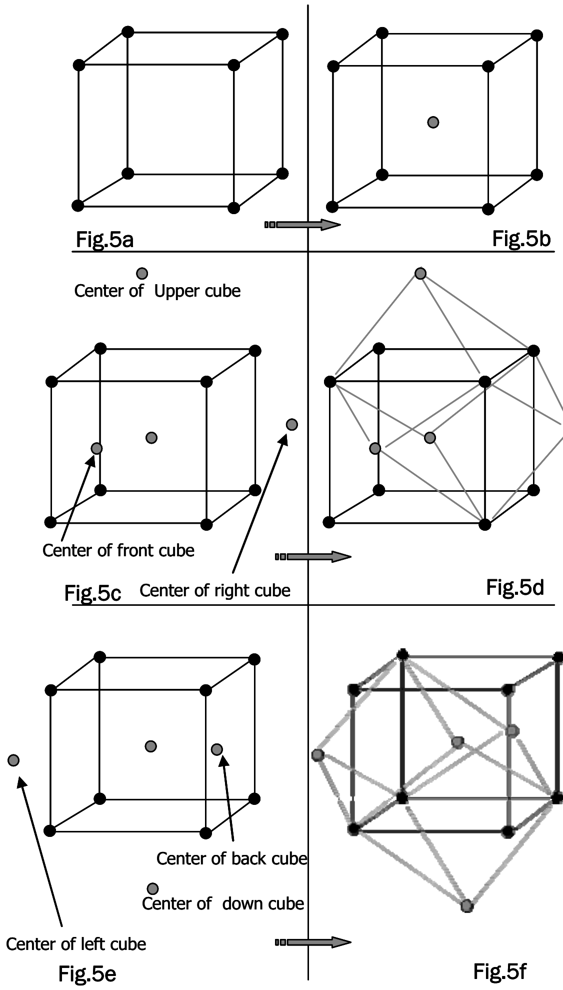


Fig. 5. Body-centered cubic structure

We demonstrate the preexisting MANET nodes with black color, while the new arriving nodes are in gray color. Figure 6 demonstrates the BCC algorithm and the process that initiates from an existing MANET cube and concludes to two new cubes that both have the central node (body-centered cubic node) of the initial MANET cube as common node.

Case 2: Extracting nodes from an established MANET

The process for extraction of nodes from an established MANET is similar to the addition of new nodes to such a MANET. The remaining nodes of a simple cube find close cubes and perform the BCC algorithm. This process changes only the global session key of the MANET, since according to BCC algorithm cube leaders have to establish a new global key; however, the simple cube session keys of the rest of the MANET cubes remain unchanged. Figure 7 demonstrates the simple case where one node leaves an established cube (node 100 of the left cube of Fig. 7a). The remaining seven nodes take place according to the BCC algorithm at the centers of the geometrically neighboring cubes of the right cube of Fig. 7a, as shown in Fig. 7b. The BCC algorithm concludes with the generation of two new cubes as shown in Fig. 7c.

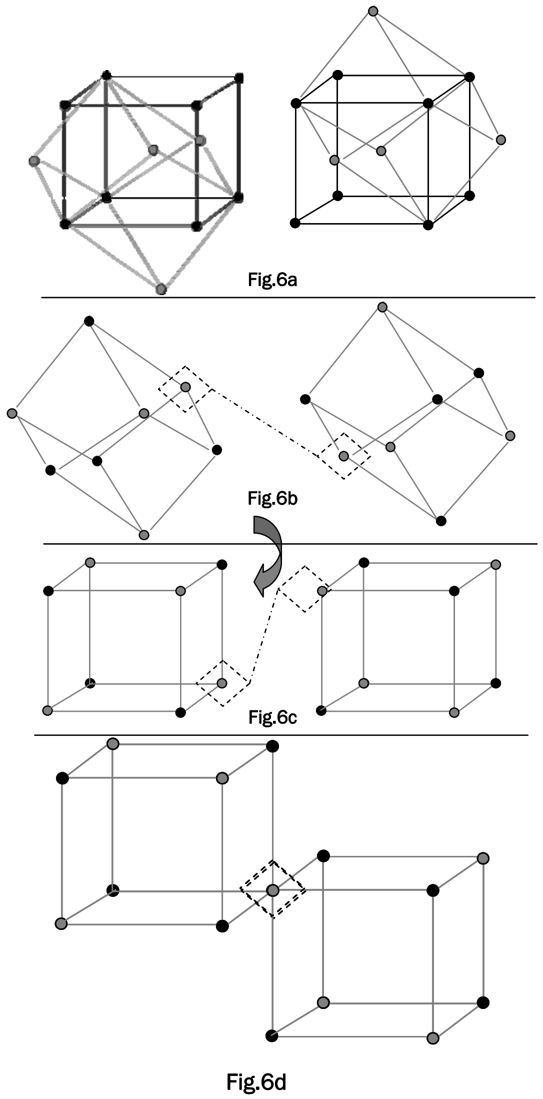


Fig. 6. Body-centered cubic algorithm

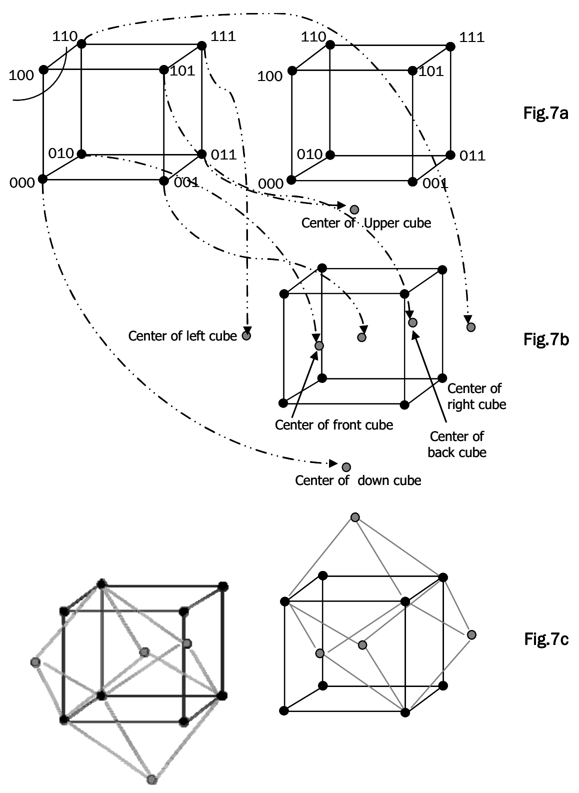


Fig. 7. Extracting nodes from an established MANET

4. Analysis of BCC approach

So far we have discussed specific problems of MANETs by focusing on the security of the network that has no supporting infrastructure. Our proposal integrates elliptic curve cryptography together with the password-based DH key exchange protocols and extends this approach to the multiparty case where new network nodes arrive or existing network nodes depart from the MANET.

The static multiparty case is considered and after careful examination the aggressive d-cube algorithm is deployed in the 3-d case. The algorithm [15] is a new version of the d-cube algorithm proposed in [14], with additional features that enhance the resistance against the dictionary attacks [1]. The basic idea behind the algorithm's design is to isolate the faulty nodes in the earliest possible stage. We managed to reduce the interaction with the faulty nodes and therefore minimize the exposure to dictionary attacks and other types of attacks. However, the aggressive behavior of the algorithm may lead to isolation of even legitimate members due to reasons such as the loss of commutation signal or the typo errors of the initial password during the authentication procedure. To overcome these kind of problems and to provide a more dynamic and robust solution, we propose the body-centered cubic (BCC) algorithm.

The BCC algorithm can be considered as a series of different aggressive cube algorithms, each one performs for a different set of MANET nodes. In terms of two-party ECDH procedures, the BCC complexity is not higher than any of the single aggressive cube algorithms. This is because within a single two-party ECDH procedure of the BCC algorithm, three ECDH procedures run simultaneously, each one corresponding to a different round of the aggressive cube algorithm.

5. Conclusions

We have introduced a new key establishment method that can overcome existing problems in this area such as rapid deployment, accuracy, and dynamic and robust behavior. We have managed to create a stable and secure structure originating from a weak secret. We have reviewed previous researches about two-party or multiparty authentication and key exchange and we have introduced the use of elliptic curve cryptography in such a scenario.

ECC computations require less storage, less power, less memory, and less bandwidth than other systems. This allows implementation of cryptography in constrained platforms such as wireless devices, handheld computers, smart cards, and thin-clients. For a given security level, elliptic curve cryptography raises computational speed and this is important in ad hoc networks, where the majority of the clients have limited resources.

We have also described known protocols for password authenticated multiparty DH key exchange and have chosen integrated aggressive cube algorithm due to its resilience against dictionary attacks. The proposed protocol retains all the good properties of its initial specification and is stronger in terms of security. Finally, we have proposed a new faster protocol for the dynamic case, where the composition of the ad hoc network changes in time with the arrivals and departures of nodes.

The initial goal of our research was to provide solutions to military operations. The key consideration was that during a military operation, arriving groups of soldiers and officers need to set up their systems quickly, efficiently, and securely, without relying on preexistent infrastructure. Due to the dynamic nature of such operations, the ability to add or subtract nodes to the established MANET is crucial. The latter may be more important since the case of captured nodes by the enemy should be considered. The secure dynamic recomposition of the network could be proved very useful in battlefields where a soldier, under threat of capture, signs off the network on time.

The proposed BCC algorithm can be applicable in several other scenarios such as emergency situations, where rescue workers arrive at a disaster field, or for groups of people meeting in a room, i.e., in a classroom together with the teacher, etc. The password-based feature of our work could be used in cases where a group of people meets one another in person for the first time, and would like to go back home and set up a secure network among them.

The proposed algorithm leaves several open issues for future work. Formal analysis is necessary. The incorporation of several new password-based key establishment protocols, which do not require the use of asymmetric encryption, is a challenging consideration. The case where the number of network entities fluctuates unevenly, changing the network topology rapidly, is also very interesting.

References

[1] Arvind Narayanan, Vitaly Shmatikov, Fast dictionary attacks on passwords using time-space trade-off Conference on Computer and Communications Security, *Proceedings of the 12th ACM conference on Computer and communications security*, Alexandria VA USA, 2005

[2] Steven M. Bellovin and Michael Merrit, Encrypted key exchange: Password based protocols secure against dictionary attacks, *Proceedings of the IEEE Symposium on Research in Security and Privacy*, Oakland USA May 1992.

[3] W. Diffie and M.E. Hellman, New directions in cryptography, *IEEE Transactions on Information Theory*, *22: 644–654*, 1976.

[4] F. Cucker and S. Smale. Complexity estimates depending on condition and round off error. *Journal of the Association for Computing Machinery, 46(1):113–184*, 2000.

[5] N. Koblitz, Elliptic curve cryptosystems, *Mathematics of Computation, volume 4 8, 203–209*, 1987.

[6] R. Rivest, A. Shamir, and L. M. Adleman, A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM, 21(2): 120–126*, Feb 1978.

[7] A. Menezes, T. Okamoto and S. Vanstone, Reducing elliptic curve logarithms to logarithms in a finite field, *IEEE Transactions on Information Theory 39:1639–1646*, 1993.

[8] A. Menezes, E. Teske, and A. Weng. Weak Fields for ECC. In T. Okamoto, editor, Topics in *Cryptology CT-RSA 2004, volume LNCS 2964, pages 366{386. Springer*, 2004.

[9] D. Johnson and S. Vanstone, The elliptic curve digital signature algorithm (ECDSA) (with D. Johnson and S. Vanstone) *International Journal on Information Security*, *1:36–63*, 2001.

[10] A. A. Kalele, V. R. Sule, *Weak keys of pairing based Diffie Hellman schemes on elliptic curves*, Cryptology ePrint Archive 2005/30, 2005

[11] Dong Zheng, Kefei Chen, Jinyuan You**,** Multiparty authentication services and key agreement protocols with semi-trusted third party, *Journal of Computer Science and Technology archive, Volume 17 , Issue 6 (November 2002) table of contents Pages: 749– 756*, 2002

[12] Giuseppe Ateniese, Michael Steiner, and Gene Tsudik, New Multiparty Authentication Services and Key Agreement Protocols, *IEEE Journal of Selected Areas in Communications, 18(4)*, April 2000

[13] Claus Becker and Uta Wille, Communication complexity of group key distribution, in *5th ACM Conference on Computer and Communications Security*, San Francisco, California, Nov.1998.

[14] N. Asokan, P Ginzboorg, Key agreement in ad hoc networks*, Computer Communications, 23:1627–1637*, 2000

[15] I.G. Askoxylakis, D.D. Kastanis and A.P. Traganitis, Elliptic curve and password based dynamic key agreement in wireless ad-hoc networks, *Communications, Networks and Information Security CNIS-2006*, Cambridge USA Oct.2006

[16] Lijun Liao and Mark Manulis,Tree-Based Group Key Agreement Framework for Mobile Ad-Hoc Networks. *Future Generation Computer Systems (FGCS),23(6):787–803,* 2007*, Elsevier*