HAL
archives-ouvertes.fr

Security of the Multiple-Key Blom's Key Agreement Scheme for Sensor Networks

Mee Yang, Adnan Anbuky, William Liu

▶ **To cite this version:**

Mee Yang, Adnan Anbuky, William Liu. Security of the Multiple-Key Blom's Key Agreement Scheme for Sensor Networks. Nora Cuppens-Boulahia; Frédéric Cuppens; Sushil Jajodia; Anas Abou El Kalam; Thierry Sans. 29th IFIP International Information Security Conference (SEC), Jun 2014, Marrakech, Morocco. Springer, IFIP Advances in Information and Communication Technology, AICT-428, pp.66-79, 2014, ICT Systems Security and Privacy Protection. <10.1007/978-3-642-55415-5_6>. <hal-01370354>

HAL Id: hal-01370354
https://hal.inria.fr/hal-01370354

Submitted on 22 Sep 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution| 4.0 International License

Security of the Multiple-Key Blom's Key Agreement Scheme for Sensor Networks

Mee Loong Yang[1], Adnan Al Anbuky[2], and William Liu[1]

[1]School of Computer and Mathematical Sciences
[2]School of Engineering
Auckland University of Technology
Auckland, New Zealand
http://www.aut.ac.nz

Abstract. The security of the Multiple-Key Blom's (MKB) key agreement scheme is analysed. We considered how the scheme may be broken by a very powerful and well resourced adversary who is able to capture any number of nodes to extract all the sensitive keying material. We showed that by choosing suitable keying parameters, the captured private keys cannot be used directly to break the scheme. Each captured key must first be correctly associated with the public key and master key used to compute it. The chances of finding this private-public-master-key association (PPMka) can be made extremely small and would require the attacker to capture a very large number of nodes, or try an extremely large number of possible solutions. This allows the scheme to be secure for use in large networks, overcoming the limitations in the original Blom's scheme. We obtained some analytical results and compared them to those from computer simulated attacks on the scheme.

1 Introduction

In our previous works [1] [2], we presented the Multiple-Key Blom's (MKB) key agreement scheme for sensor networks. This scheme is fast, efficient and frugal, making it specially attractive for low power devices in ad hoc mobile networks. In this paper we show that it is also resilient against a powerful and well resourced adversary who is able capture a large number of nodes and extract all keying material.

For ad hoc mobile networks, an identity-based cryptographic (IBC) key establishment protocol would be very useful for pairs of node to derive their pairwise keys when needed. As defined in [3] an IBC key establishment protocol uses an entity's identity (*ID*) information (e.g. name and address, identifying index, etc.) as its public key. While its origin is usually attributed to Shamir [4] where the *ID* can be the node's name and address, according to Menezes, Oorschot & Vanstone [3], Blom was first to propose the identity-based (or more accurately, index-based) key establishment scheme in [5][6]. The Blom's scheme is unconditionally-secure in the information-theoretic sense if less than a certain number of nodes are compromised. Once this threshold is exceeded, the scheme

can be completely broken. Recognising this limitation, Blom [5] said "It would be nice to have systems that degrade more gracefully but more research is needed". We believe that our scheme is able to fulfil this requirement.

This Contribution We considered the security of the Multiple-Key Blom's scheme in the three aspects – the strength of the keys, the security of the underlying Blom's scheme as it applies to our scheme, and the probabilities of the scheme being completely broken by a very powerful adversary. We presented analytical results to compute the probability of success in breaking the scheme and compared it with computer simulated attacks on some implementations.

The paper is structured as follows: In Section 2, we described briefly some related works and the necessary background material. Section 3 dealt with the security of the keys and how our scheme would improve the security of the original Blom's scheme. In Section 4 we presented some analytical results on the effort required and the probabilities of breaking the scheme. We gave our conclusions in Section 5.

Notations and terms used

$\mathbf{K}$ – private key, a secret $(1 \times m)$ row vector unique to the node
$\mathbf{M}$ – master key, an $(m \times m)$ secret symmetric matrix
N – number of master keys
R – pairwise key-set, the set of numbers used to form the pairwise key
$\mathbf{V}$ – public key, a $(m \times 1)$ column vector unique to the node
m – the size of the master key matrix
η – number of public keys assigned to each node
p – prime modulus for key operations
q – prime modulus for public key operations only
s – public key seed, an integer $\in [0, q-1]$

2 Related Works

The original Blom's scheme has limitations in that to be secure, it would require substantial memory for storage of keying material when used in large networks. The work in [7], improved on its scalability by using a clustered topology where only the cluster-heads implemented the Blom's scheme. The probabilistic scheme in [8] used multiple key-spaces where nodes would first discover their shared key space to implement the scheme. A recent work in [9] added some constrained random perturbations to the private keys to break its direct relationship with the master key, thereby increasing the resilience against the master key being computed from sufficient number of stolen private keys. All these works did not address the issue of the pairwise key sizes which would be same as the data size used for the master key elements.

The BYka scheme

The basic concepts of our Multiple-key Blom's key agreement scheme, now called the Blom-Yang key agreement (BYka) scheme, has been presented in [1] [2]. Due to space constraint only a brief description suffice for this paper is given here.

Trusted Authority and Master keys The Trusted Authority (TA) is responsible for all keying material. It generates N secret master keys $\mathbf{M}_1, \mathbf{M}_2, \cdots, \mathbf{M}_N$, each one being a random $(m \times m)$ symmetric matrix $\mathbf{M}$ over the prime field $\mathbb{F}_p$.

Public key-tag (*ID*) The TA assigns each node one set of public keys called the "public key-set" consisting of η column vectors over the prime field $\mathbb{F}_q$, where $q \gg p$. For example, for node A, the public key-set is $\{\mathbf{V}_{A_1}, \cdots, \mathbf{V}_{A_\eta}\}$. These vectors are columns of the Vandermonde matrix, i.e.,

$$\mathbf{V}_{A_i}^T = \left[1 \; s_{A_i} \; s_{A_i}^2 \; \cdots \; s_{A_i}^{m-1}\right] \pmod{q}, \text{ for } i = 1, \cdots, \eta$$

The values s_{A_i} are called the public key "seeds". The seeds $s_{A_1}, \cdots, s_{A_\eta}$ are consecutive such that the smallest seed, s_{A_1} is a multiple of η. In this way, each node's public key-set is unique and can be concisely represented by just the smallest seed. This serves as the node's identity *ID*, also called the "public key-tag". For example, node A's public key-set can be represented by its public key-tag $ID_A = s_{A_1}$.

Private keys The TA computes the private keys for each node using all permutations of their η public keys with the N master keys. For node A, the private keys are a set of ηN $(1 \times m)$ row vectors, called the "private key-set" computed as follows,

$$\mathbf{K}_{A_{ij}} = \mathbf{V}_{A_i}^T \mathbf{M}_j \pmod{p}, \quad \text{for } i = 1, \cdots, \eta, \; j = 1, \cdots, N$$

Consider the u^{th} element of the private key $\mathbf{K}_{A_{ij}}$,

$$\begin{aligned} K_{A_{ij_u}} &= \sum_{n=1}^{m} s_{A_i}^{n-1} \pmod{q} \; M_{j_{nu}} \pmod{p} \\ &= M_{j_{1u}} + s_{A_i}^1 M_{j_{2u}} + \cdots + s_{A_i}^{m-1} M_{j_{mu}} \pmod{p} \end{aligned} \tag{1}$$

The public key operations are modulo q, while all other key operations are modulo p. It is possible for multiple public keys to map to the same key in Eqn. (1), a phenomenon we call "key aliasing".

To prevent key aliasing, a seed s is chosen such that at least one vector element is $> q$ and $\not\equiv 0 \pmod{p}$, i.e.,

$$\left.\begin{array}{r} \text{for some } \; w \leqslant m, \quad s^{w-1} > q \\ \text{i.e. } \; s^{w-1} \equiv r \pmod{q} \\ \text{and} \quad r \not\equiv \begin{cases} 0 \pmod{p}, & \text{or} \\ s \pmod{p} \end{cases} \end{array}\right\} \tag{2}$$

The TA installs into each node their keying material comprising the global keying parameters $\{m, N, \eta, p, q\}$, the node's individual public key-tag ID, and its private key-set $\mathbf{K}_{1,\cdots,N\eta}$. Crucially, the private keys in the key-set are stored in a random order. All these are static and can be stored in the ROM or flash memory.

Pairwise Key Derivation After deployment, any pair of nodes can derive their common secret pairwise key after exchanging their IDs, a very small amount of bits. For example, A and B have obtained each other's IDs. Next, each node generates their counterpart's public keys. For example, node A generates B's public keys,

$$\left.\begin{array}{r} s_{B_k} = ID_B + (k-1), \\ \mathbf{V}_{B_k}^T = \left[1 \; s_{B_k} \; s_{B_k}^2 \cdots s_{B_k}^{m-1}\right] \pmod q \\ \text{for} \;\; k = 1, \cdots, \eta \end{array}\right\} \tag{3}$$

Then, using all the permutations with its own private keys, the nodes computes (modulo p), the "pairwise key-sets" R_A and R_B as follows,

$$\begin{array}{rc} \text{Node A:} & R_{A_{ijk}} = \{\mathbf{K}_{A_{ij}} \mathbf{V}_{B_k}\} = \{(\mathbf{V}_{Ai}^T \mathbf{M}_j)\mathbf{V}_{B_k}\} \\ \text{Node B:} & R_{B_{ijk}} = \{\mathbf{K}_{B_{ij}} \mathbf{V}_{A_k}\} = \{(\mathbf{V}_{Bi}^T \mathbf{M}_j)\mathbf{V}_{A_k}\} \\ \multicolumn{2}{c}{\text{for} \;\; i, k = 1, \cdots, \eta, \;\; \text{and} \;\; j = 1, \cdots, N} \end{array}$$

Transposing all the elements in the set R_B we have,

$$R_{B_{ijk}} = \{((\mathbf{V}_{B_i}^T \mathbf{M}_j)\mathbf{V}_{A_k})^T\} = \{(\mathbf{V}_{A_k}^T \mathbf{M}_j^T)\mathbf{V}_{B_i}\}$$

Since $\mathbf{M}_j$ is symmetric, $\mathbf{V}_{Ai}^T \mathbf{M}_j \mathbf{V}_{B_k}$ is scalar, and i, j, k are merely independent counters, the sets R_A and R_B each contain $N\eta^2$ identical numbers, though not in the same order. These numbers would be used by both nodes to form a pairwise key using a preconfigured method.

In our scheme, the sequence formed from the number of occurrences of the integers $0, 1, \cdots, p-1$ is used as input to a hash function to obtain a 128-bit pairwise key.

3 Security of the BYka scheme

3.1 Security Model

System Model The system comprises nodes belonging to one administrative unit under the TA. The TA has access to a good random number generator for generating the master key matrices. Before deployment, each node uses a secure connection with the TA to obtain its keying material.

The nodes have very limited computing power, memory, and battery life. They have access to strong cryptographic services such as hash functions, pseudo random number generators, and strong symmetric encryption techniques such as the AES algorithm. They are highly mobile, are deployed in an ad hoc manner, and communicate with each other using low power radio with a limited range.

Adversary model The adversary is a very powerful agent capable of moving about freely in the deployment space to monitor and insert messages. In addition, it is capable of capturing any number of nodes to extract all the keying material including the public and private keys from ROM and RAM. It also has access to unlimited computing resources. It cannot compromise the TA.

System breakdown The scheme is consider broken if the adversary is able to, by monitoring messages and/or obtaining sensitive information from captured nodes, compute the pairwise keys of any pair of un-compromised nodes, fabricate new valid public and private keys, or compute the master keys of the TA. Identity theft attacks are not considered in this paper.

3.2 Vulnerabilities

The vulnerabilities of the BYka scheme can be studied in three main aspects:

1. Resistance of the keys against brute force attacks,
2. Security of the Blom's scheme on which is is based, and
3. Resilience against node capture

3.3 Security of keys against brute force

Master keys Each master key is an $m \times m$ symmetric matrix. It has $\frac{m(m+1)}{2}$ unique elements, each one being a random number $\in [0, p-1]$. The brute force attacker would have to try all the $p^{\frac{m(m+1)}{2}}$ possible keys. For example, even with small values of $m = 12,\ p = 13$, there are 7.72×10^{86} possible keys, equivalent to 288 bits.

Private keys In Eqn. (1), the elements of the private keys, being products and sums of random numbers, are also random. Hence, each private key is just a row vector of random numbers and is indistinguishable from each other.

There are $\eta N m$ elements in each private key-set. Even with small values of $m = 12, N = 6, \eta = 6$ and $p = 13$, there are about 1.673×10^{481} possible keys, or equivalent to $1,599$ bits. This is large enough to defeat the brute force attempt to fabricate a node's private key-set.

Pairwise key The BYka scheme can be viewed as a mechanism for two nodes to derive identical (unordered) key-sets R_A and R_B which contains $N\eta^2$ integers $\in [0, p-1]$. The numbers in the key-set, for example, $R_{A_{ijk}} = \mathbf{K}_{A_{ij}} \mathbf{V}_{B_k} = \sum_{n=1}^{m} K_{A_{ij}} s_{B_k}^{n-1} \pmod p$, where s_{B_k} satisfy Eqn. (2), are also random numbers $\in [0, p-1]$. Hence the number of occurrences of the integers $1, 2, \cdots, p-1$ in the key-set R_A is also random.

Pairwise key size The pairwise-key set contains $N\eta^2$ integers $\in [0, p-1]$. These can be combined together to form a pairwise key of up to $N\eta^2 b$ bits, where b is the data size of p. For example with $N, \eta = 6$, and $p = 31$, the key size can be up to 1080 bits.

Pairwise keyspace The number of possible pairwise keys, or the keyspace size, must be at least as large as the desired key size. It is however, limited by the number of possible combinations of the $N\eta^2$ integers in the key-sets R. The keyspace size can be determined by considering the number of possible combinations of the integers $1, 2, \cdots, p-1$, such that the total number of integers in each combination is exactly $N\eta^2$. This can be obtained by considering the following partitioning problem.

Given a row of $N\eta^2$ items, we wish to partition them such that there are p groups, $g_0, g_1, \cdots, g_{p-1}$, each containing the integers $0, 1, \cdots, p-1$ respectively. To create the partitions, we first insert $(p-1)$ items into the row so that there are now $(N\eta^2 + p - 1)$ items. If any $(p-1)$ items are now removed, it would leave $(p-1)$ gaps separating the remaining items into p groups as desired. The number of ways to remove $(p-1)$ items from $(N\eta^2 + p - 1)$ gives the keyspace size as follows,

$$K_{space} = \binom{N\eta^2 + p - 1}{p - 1} \tag{4}$$

Using suitable values of N, η, and p, keyspace sizes of 64, 80, 96, and 128 bits are possible, as shown in Table (1).

η	N	Values of p				
		13	17	19	23	31
6	6	64	80	88	102	127
	7	67	84	92	106	134
	8	69	87	95	111	139
7	6	69	87	95	111	140
	7	72	91	99	116	146
	8	74	94	103	120	152
8	6	74	93	102	119	151
	7	77	97	106	124	157
	8	79	100	109	128	163

Table 1. Key space sizes in bits

3.4 Security of the underlying Blom's scheme

In our previous work [2], we showed how the Blom's scheme would be completely broken if the number of nodes compromised is m, the "capture threshold". The scheme is said to be unconditionally secure if no more than $(m-1)$ nodes are compromised, and all the public key vectors are linearly independent of each

other [3]. Otherwise, using the captured private keys from m or more nodes, the attacker would be able to either mount the Sybil attack by fabricating new public and private keys using linear combinations of the m captured keys, or attack the master key $\mathbf{M}$ by solving the system of $m \times m$ linear equations, $\mathbf{K}_X = \mathbf{V}_X^T \mathbf{M}$.

The security of the BYka scheme would appear to be similar to the original Blom's scheme. Apparently, the capture threshold is lower at $\lceil \frac{m}{\eta} \rceil$ since each node has η private keys associated with each master key. However, the attacker would first have to associate each private key with the public key and master key used to compute it, i.e. discover the private-public-master-key associations (PPMka).

Each private key computed in Eqn. (1) is a row vector of random integers $\in [0, p-1]$ and has no information about the public key and master key used to compute it. Without the PPMka information each key can be correctly associated with the public key and master key with a probability of $\frac{1}{N\eta}$. To mount the Sybil or master key attack, all the m private keys must be used together. This will result in a very large number of possible solutions as shown later.

4 Attacks to discover the PPMka

4.1 Using brute force

Using one captured node Consider that the attacker has obtained the public and private keys from one captured node. The attacker generates an arbitrary master key, and using one of the public keys, computes a trial private key using Eqn. (1). This is then compared with each of the captured keys to find a match. After trying all the possible master keys, there will be N matches and eventually all the master keys will be found. The number of possible master keys to try is $p^{\frac{m(m+1)}{2}}$ which is a very large number with typical keying parameters.

Using sufficient captured nodes Consider that $\frac{m}{\eta}$ nodes have been captured. Each node has $N\eta$ private keys. There is enough information to construct the N systems of $m \times m$ equations to solve for all the master keys, using for example, the following procedure.

From each node, the $N\eta$ keys are grouped into $\binom{N\eta}{\eta}$ possible groups each associated with one of the master keys. Within each group, each private key can be associated with the η public keys in $\eta!$ ways. Therefore for each node there are $\binom{N\eta}{\eta}\eta!$ ways to obtain the set of ηm equations for solving one master key, say $\mathbf{M}_1$. Using a total of $\frac{m}{n}$ nodes, it is possible to obtain a set of $m \times m$ equations to solve for $\mathbf{M}_1$.

After obtaining the first master key, the η private keys associate with $\mathbf{M}_1$ is removed and the process is repeated for $\mathbf{M}_2$, and so on. Overall, the total number of attempts required to obtain all the master keys is

$$\Phi = \sum_{i=0}^{N-1} \left[\binom{N\eta - i\eta}{\eta} \eta! \right]^{\frac{m}{\eta}}$$

The number of iterations required, even with small parameter values is very large. For example with $m = 12,\ N = 6,\ \eta = 6$, there are 2.16×10^{18} possible master keys solutions.

4.2 Pairing Attack to discover the PPMka

A better approach would be to get pairs of nodes, e.g. nodes A and B, to compute their pairwise key-sets R_A and R_B using each other's public keys. The $N\eta^2$ numbers in the two key-sets will be identical, and if they are unique, the attacker would be able to, by matching them, associate the related private keys to the same master key. We call this the "pairing attack".

A more efficient pairing attack would use only one of each other's public keys to compute the partial key-sets R_{rA} and R_{rB} which now contains only $N\eta$ elements. This is illustrated in Fig. (1) for the simple case using $N, \eta = 2$. Here, since $\mathbf{K}_{A_1}\mathbf{V}_{B_2} = \mathbf{K}_{B_3}\mathbf{V}_{A_1}$, both must be associated with the same master key say, $\mathbf{M}_1$. The PPMka for the private keys can then be found, i.e. $\mathbf{K}_{A_1} = \mathbf{V}_{A_1}^T\mathbf{M}_1$, $\mathbf{K}_{B_3} = \mathbf{V}_{B_2}^T\mathbf{M}_1$ and similarly, $\mathbf{K}_{A_2} = \mathbf{V}_{A_1}^T\mathbf{M}_2$, $\mathbf{K}_{B_2} = \mathbf{V}_{B_2}^T\mathbf{M}_2$.

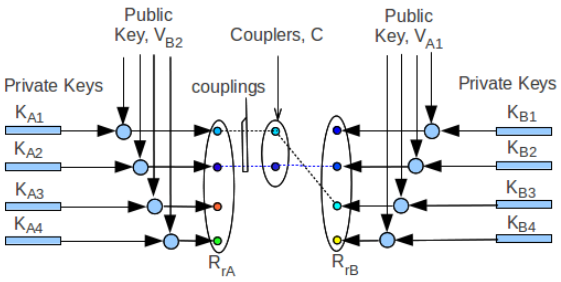

Fig. 1. Pairing attack for case $N = 2, \eta = 2$

Collisions If all the numbers in R_r are unique, the above attack would be successful. However, if they are not, we say there are "collisions", and there are more than one possible PPMka's for the affected private key.

Couplers and Couplings We call the numbers that are identical across both partial key-sets R_{rA} and R_{rB}, "couplers". In Fig. (1), the set C contains the couplers. The links connecting the couplers to the numbers in R_{rA} and R_{rB} are called "couplings". The number of couplings linking R_r to the couplers is denoted as N_c.

In the ideal case where there is no collision, there would be exactly $N_c = N$ couplings on each side of C, each one linking the private key to the associated master key and public key, revealing their PPMka. However, if the couplers are not distinct, then the PPMka's for the related private keys are ambiguous.

The probability of having all distinct couplers in the partial key-set of $N\eta$ numbers is $P_u = (\frac{p}{p})(\frac{p-1}{p})\cdots(\frac{p-N\eta-1}{p})$. This can be made very small by choosing suitable values of p, N and η. For example, with $p = 31,\ \eta = 6,\ N = 5$, we have $P_u = 2.49 \times 10^{-12}$. If $N\eta \geqslant p+1$, the probability P_u is zero as there is not enough numbers to go round without repetition.

Pairing Attack approaches We consider two extreme approaches to discover the PPMka. In the first case, the "unlimited capture" attack, the attacker is able to capture as many nodes as necessary until finally the PPMka can be exposed. At the other end of the spectrum, in the "limited capture" attack, the attacker has just enough nodes to compute the master keys.

4.3 Unlimited Capture Attack

Traitor Node The pairing attack would be successful if each pairing results in key-sets R in which all the numbers are unique. However with suitable choice of keying parameters this probability is very small. Nevertheless, the attack would have a better chance of success if a node is available such that all the N private keys associated with one of the public keys is known or "exposed". This set of private keys can be used to reduce the ambiguities in subsequent pairings. We call this node the "traitor node" since it can be used to betray other nodes. For example in Fig.(1), nodes A and node B are possible traitor nodes.

In general, a traitor node T is found if, in a pairing, the number of couplings it has is $N_c = N$ i.e. there are $\leqslant N$ couplers. If $N_c > N$, there will be ambiguities.

Using the traitor node, another node say B, is paired with it. If the number of couplings in R_{rB} is N, they distinctly link the related private keys in B to the exposed private keys in T revealing the PPMka. For example, in Fig. (2), T is a traitor node and the keys $\mathbf{K}_{T1}$ and $\mathbf{K}_{T2}$ are known to be associated with $\mathbf{M}_x$ and $\mathbf{M}_y$ respectively. Then for node B, the keys $\mathbf{K}_{B1}$ and $\mathbf{K}_{B2}$ must be associated with $\mathbf{M}_x$ and $\mathbf{M}_y$ respectively, and both associated with public key $\mathbf{V}_{B2}$.

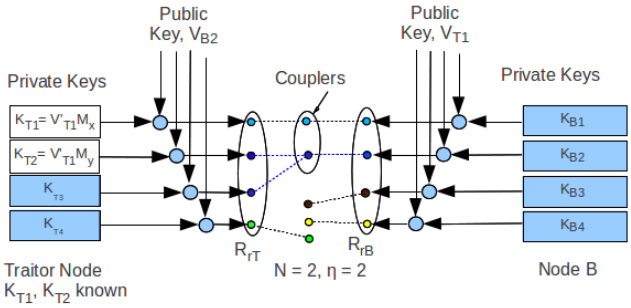

Fig. 2. Traitor Node can be used to attack the PPMka

This is not so straightforward if the number of couplings or couplers in R_{rB} is $\neq N$. The PPMka of the keys related to colliding couplers will be ambiguous, as in Fig. (3). Fig. (3a) shows the partial key-set R_{rB} having only 1 coupler. While $\mathbf{K}_{B1}$ and $\mathbf{K}_{B2}$ can both be associated with $\mathbf{V}_{B2}$, their associations with the master keys are ambiguous. In Fig. (3b), R_{rB} has more than N couplings, i.e. 3 instead of 2. Now it not clear whether $\mathbf{K}_{B2}$ or $\mathbf{K}_{B3}$ is associated with $\mathbf{V}_{B2}$ and master key $\mathbf{M}_y$.

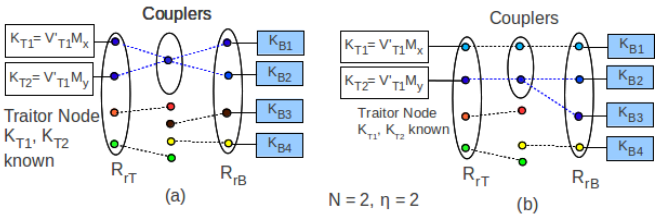

Fig. 3. Traitor Node cannot be used to discover the PPMka

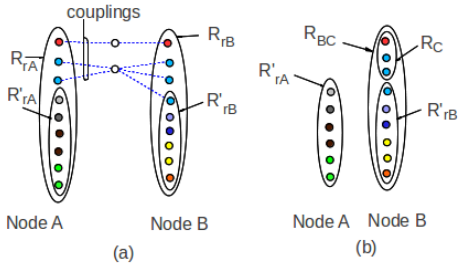

Fig. 4. Partial key-sets

Probability of finding a traitor node To determine the probability of finding a traitor node, we can consider the following problem. In the Fig. (4a), the pairing attack produces partial key-sets R_{rA} and R_{rB}. We remove the couplers from R_{rA} to form the set R_c, leaving the reduced partial key-sets R'_{rA} and R'_{rB} in Fig. (4b). A traitor node is found if the reduced set R'_{rA} is disjoint with $(R'_{rB} \cup R_c)$, or R'_{rB} is disjoint with $(R'_{rA} \cup R_c)$. In addition, the sets R'_{rA}, R'_{rB} and R_c can all be disjoint.

The probability of these occurrences can be found by counting the number of arrangements for the above cases. Let N_a, N_b and N_c be the number of elements in sets R'_{rA}, R'_{rB} and R_c respectively. Here, $N_a = N_b = N\eta - N$ and $N_c = N$. The number of elements in $(R'_{rB} \cup R_c)$ is $N\eta$.

First, consider the general case of arranging N_a numbers given r numbers, such that each arrangement has **all** the r numbers. For example, in arranging 4 numbers given the 3 numbers $\{6, 7, 8\}$, permutations like $\{6, 6, 7, 8\}$ and $\{6, 7, 7, 8\}$ would be included, but excludes permutations such as $\{6, 6, 6, 7\}$ and $\{6, 6, 7, 6\}$, etc. Let the number of arrangements be $Q_{N_a r}$. It can be shown that

$$Q_{N_a r} = r^{N_a} - \sum_{i=1}^{r-1} \binom{r}{i} Q_{N_a i} \quad \text{where} \quad Q_{N_a 1} = 1 \tag{5}$$

The total number of arrangements where R'_{rA} is disjoint with $(R'_{rB} \cup R_c)$ is then,

$$\theta_u = \sum_{r=1}^{N_a} \binom{p}{r} Q_{N_a r} (p - r)^{N\eta} \tag{6}$$

It is also possible that the sets R'_{rA}, R'_{rB} and R_c are all disjoint. The number of such arrangements θ_d, can be similarly shown to be,

$$\theta_d = \sum_{r=1}^{N_c} \left[\binom{p}{r} Q_{N_c r} \times \left(\sum_{k=1}^{N_a} \binom{p-r}{k} Q_{N_a k} (p-r-k)^{N_b} \right) \right] \tag{7}$$

where $Q_{N_c r}$ and $Q_{N_a k}$ can be obtain as in Eqn. (5).

The total number of possible arrangements is $\theta_t = 2\theta_u - \theta_d$, without double counting the cases for all 3 disjoint sets.

The probability of finding a traitor node is then,

$$P_t = \frac{\theta_t}{p^{2N\eta - N}} = \frac{2\theta_u - \theta_d}{p^{2N\eta - N}} \tag{8}$$

The values of P_t for various keying parameters is given in Table (2).

η	N	$p = 13$		$p = 17$		$p = 31$	
		P_t	n_c	P_t	n_c	P_t	n_c
6	6	1.07×10^{-16}	2.29×10^{7}	1.70×10^{-15}	5.75×10^{6}	5.62×10^{-12}	1.00×10^{5}
	7	5.25×10^{-19}	1.02×10^{9}	8.71×10^{-19}	2.51×10^{8}	5.01×10^{-15}	3.31×10^{6}
	8	2.57×10^{-23}	4.68×10^{10}	4.37×10^{-22}	1.12×10^{10}	3.72×10^{-18}	1.23×10^{8}
7	6	2.40×10^{-20}	1.29×10^{9}	4.07×10^{-19}	3.16×10^{8}	2.45×10^{-15}	4.07×10^{6}
	7	2.88×10^{-24}	1.20×10^{11}	5.01×10^{-23}	2.88×10^{10}	4.68×10^{-19}	2.95×10^{8}
	8	3.47×10^{-28}	1.10×10^{13}	6.17×10^{-27}	2.57×10^{12}	7.59×10^{-23}	2.34×10^{10}
8	6	5.50×10^{-24}	7.59×10^{10}	9.55×10^{-23}	1.82×10^{10}	8.71×10^{-19}	1.91×10^{8}
	7	1.62×10^{-28}	1.38×10^{13}	2.88×10^{-27}	3.31×10^{12}	3.63×10^{-23}	2.95×10^{10}
	8	4.79×10^{-33}	2.57×10^{15}	8.51×10^{-32}	6.03×10^{14}	1.32×10^{-27}	4.90×10^{12}

Table 2. Probabilities of finding a Traitor Node P_t, and expected capture sizes n_c

Expected Node capture We assume that attacker is able to capture any number of nodes. As a new node is captured, it is paired with each of the previous ones to find the traitor, if not another new node is captured and the process repeated. Since the probability of finding a traitor node is P_t, the expected number of pairing attempts to find one is $\frac{1}{P_t}$. Each node has η public keys to try, so each pair gives η^2 attempts. If the number of nodes captured is n_c, the number of pairs that can be formed is $\binom{n_c}{2}$ giving a total of $\eta^2\binom{n_c}{2}$ pairing attempts. To find a traitor node we have,

$$\begin{aligned} \eta^2 \frac{n_c!}{2!(n_c-2)!} &\geqslant \frac{1}{P_t} \\ \text{i.e.} \quad n_c &\geqslant \frac{1}{2}\left(1 + \sqrt{1 + \frac{8}{\eta^2 P_t}}\right) \end{aligned} \tag{9}$$

The values of n_c for some keying parameters is also given in Table (2). It can be seen that for these cases, thousands of nodes need to be captured just to

find a traitor node. Finding a traitor does not break the scheme. The chance of finding a node which has exactly N couplers when paired with the traitor node is as improbable as finding the traitor node itself.

4.4 Limited capture pairing attack

In this attack only $\lceil \frac{m}{\eta} \rceil$, but sufficient, number of nodes has been captured. Using the pairing attack, the partial key-set is obtained. If there are $N_c > N$ couplings due to collisions, there are N_c possible ways choose the private key related to one of the master keys, say $\mathbf{M}_1$. Using all the η public keys one at a time, the number of sets of equations from one node is $[N_c]^\eta$. To obtain the $m \times m$ equation, $\frac{m}{\eta}$ nodes are required. The number of possible solutions for the master key $\mathbf{M}_1$ is $[N_c]^m$.

After obtaining the first master key, the associated private key is removed leaving $N_c - 1$ keys to choose for solving the next master key. In total, to solve for all the master keys, the total possible number of sets of equations, i.e. the number of iterations required is, $\Phi = \sum_{i=0}^{N-1} [N_c - i]^m$.

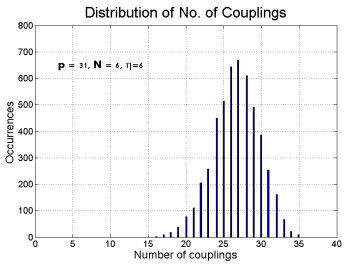

Fig. 5. Distribution of number of couplings for $p = 31, N = 6, \eta = 6$

Binomial Distribution Approximation Fig. (5) shows the typical distribution of the number of couplings in the pairing attacks, in this case for $p = 31, N = 6, \eta = 6$. It suggests that the distribution can be approximated by the binomial distribution,

$$P(X = x) = \binom{N\eta}{x} p_r^x (1 - p_r)^{(N\eta - x)} \tag{10}$$

where the x is the number of couplings, and $\mu = N\eta p_r$ is the mean. From Eqn. (8), we can compute the probability when there are N couplings, i.e. $P(X = N)$. Then solving for p_r in Eqn. (10), we can obtain the mean μ. Writing the expected number of couplings in a pairing as $N_c = \mu$, then the probable number of possible solutions for all the master keys is,

$$\Phi = \sum_{i=0}^{N-1} [N_c - i]^m = \sum_{i=0}^{N-1} [\mu - i]^m \tag{11}$$

Table (3) gives the probable number of master keys solutions, Φ for various keying parameters.

η	N	$m = 12$			$m = 24$		
		13	17	31	13	17	31
6	6	3.55×10^{17}	2.34×10^{17}	9.55×10^{16}	1.26×10^{35}	5.37×10^{34}	9.12×10^{33}
	7	2.40×10^{18}	1.66×10^{18}	7.94×10^{17}	5.75×10^{36}	2.75×10^{36}	6.17×10^{35}
	8	1.23×10^{19}	1.23×10^{19}	4.79×10^{18}	1.55×10^{38}	1.55×10^{38}	2.24×10^{37}
7	6	1.66×10^{18}	1.66×10^{18}	5.37×10^{17}	2.75×10^{36}	2.75×10^{36}	2.82×10^{35}
	7	1.23×10^{19}	1.23×10^{19}	4.79×10^{18}	1.55×10^{38}	1.55×10^{38}	2.24×10^{37}
	8	8.91×10^{19}	6.92×10^{19}	3.02×10^{19}	8.13×10^{39}	4.79×10^{39}	9.12×10^{38}
8	6	9.12×10^{18}	6.61×10^{18}	3.39×10^{18}	8.13×10^{37}	4.37×10^{37}	1.15×10^{37}
	7	6.92×10^{19}	6.61×10^{19}	3.02×10^{19}	4.79×10^{39}	2.75×10^{39}	9.12×10^{38}
	8	3.89×10^{20}	3.09×10^{20}	1.91×10^{20}	1.51×10^{41}	9.55×10^{40}	3.63×10^{40}

Table 3. Probable number of Master key solutions, Φ

4.5 Experimental Results of Simulated Pairing Attacks

A computer programme was used to implement the pairing attacks to determine the traitor capture sizes n_c and the probable number of master key solutions Φ.

The programme first generates the master keys. It then randomly creates new nodes with unique IDs to simulate captured nodes. Each node is paired with each of the previously "captured" nodes until a traitor node is found. At the same time the number of couplings is accumulated for the first $\frac{m}{\eta}$ nodes. This is the probable number of couplings in the limited captured attack. When a traitor node is found, a new implementation is done using a new set of master keys and this is repeated for 1000 times.

These are real attacks on real systems as the public and private keys can be implemented in real sensor nodes. They are "simulated" in the sense that capturing the nodes and extracting the keys are done in the computer programme, greatly accelerating the attacks.

Due to the large traitor capture sizes, only cases which gives results within a reasonable time is given in Table (4). These results are the mean values for 1000 implementations for each case. As an indication, one run for the case using $N = 6,\ \eta = 4$ took over $2\frac{1}{2}$ hours to find a traitor node, requiring about 135 captured nodes.

5 Conclusion

The security of our BYka key agreement scheme was analysed in term of the resistance of its keys against brute force attacks, the security of the Blom's scheme on which it is based, and its resilience against node capture. We showed that the keys are random and large enough to resist brute force attacks. The scheme inherits the unconditionally security of the original Blom's scheme and also allows it to break free from the limitations therein. This is because, by using multiple keys in permutations, the relationships of the private keys with the master keys and public keys becomes indiscernible, and they cannot be used

η	N	Traitor Capture size n_c		Number of solutions, Φ	
		Eqn.(9)	Expt.	Eqn.(11)	Expt.
4	4	5.59	5.23	7.97×10^{22}	1.06×10^{24}
	5	23.23	21.48	5.43×10^{26}	8.46×10^{26}
	6	128.05	113.53	7.92×10^{28}	1.10×10^{29}
5	4	24.45	21.37	7.95×10^{26}	8.16×10^{26}
	5	237.99	215.63	7.93×10^{28}	9.95×10^{29}
6	3	10.76	9.62	1.00×10^{24}	1.17×10^{25}
	4	155.91	135.88	1.68×10^{28}	1.42×10^{29}
7	3	37.57	33.04	7.95×10^{25}	7.17×10^{26}

Table 4. Comparison between analytical and experimental results for $p = 31$

directly to break the scheme. We calculated the probabilities of discovering the correct private-public-master-key association information and showed that, with suitable keying parameters, these probabilities are very small. Consequently, to break the scheme, it would require a very large number of nodes to be captured, or an infeasibly large number of solutions are possible. Our analytical results were verified by comparing them with results obtained from simulated attacks on the scheme using a computer programme. Our BYka scheme is thus secure against very powerful adversaries and being fast, efficient and frugal, would be useful for large ad hoc mobile sensor networks.

References

1. M. L. Yang, A. Al-Anbuky, and W. Liu, "A Fast and Efficient Key Agreement Scheme for Wireless Sensor Networks," *International Conference on Wireless and Mobile Communications, Venice.*, pp. 231–237, 2012.
2. ——, "The Multiple-Key Blom's Scheme for Key Establishment in Mobile Ad Hoc Sensor Networks," *The 19th Asia-Pacific Conference on Communications, Bali, Indonesia*, pp. 422–427, 2013.
3. A. J. Menezes, P. C. Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, 2001.
4. A. Shamir, "Identity-based Cryptography and Signature Schemes," *Proceedings of CRYPTO'84, LNCS 196*, pp. 47–53, 1985.
5. R. Blom, "Non-Public Key Distribution," *Advances in Cryptology, Proceedings of Crypto '82*, pp. 231–236, 1983.
6. ——, "An Optimal Class of Symmetric Key Generation Systems," Linkopping University, Tech. Rep., 1984.
7. N. Chen, J.-b. Yao, and G.-j. Wen, "An Improved Matrix Key Pre-distribution Scheme for Wireless Sensor Networks," *International Conference on Embedded Software Systems*, p. 4045, 2008.
8. W. Du, S. Y. Han, J. Deng, and P. K. Varshney, "A pairwise key pre-distribution scheme for wireless sensor networks," *Proceedings of the conference on Computer and communications security*, October 2003.
9. C.-M. Yu, C.-S. Lu, and S.-Y. Kuo, "Noninteractive Pairwise Key Establishment for Sensor Networks," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 3, pp. 556–569, 2010.