KEY-POLICY ATTRIBUTE-BASED ENCRYPTION SCHEME FOR GENERAL CIRCUITS

Diana BOLOCAN

Department of Computer Science
Alexandru Ioan Cuza of Iași, Romania
e-mail: dianabolocan.db@gmail.com

Abstract. Key-policy attribute-based encryption (KP-ABE) schemes are very important in recent applications such as cloud technology. The most efficient KP-ABE schemes known so far are based on bilinear maps and secret sharing and work for Boolean formulas. Unfortunately, these schemes loose their efficiency when applied to arbitrary Boolean circuits. The main drawback is the backtracking attack that occur at OR gates with fanout greather than one. This paper proposes a new KP-ABE scheme based on bilinear maps and secret sharing, aimed to work well for the entire class of Boolean circuits. The scheme keeps the size of the circuit unmodified, replacing the OR gates with NAND gates and adding additional functionalities in the algorithm for the new logic NAND gates. The scheme is secure under the decisional bilinear Diffie-Hellman assumption.

Key Words. *KP-ABE scheme, general circuit, decisional bilinear Diffie-Hellman assumption*

1. INTRODUCTION

Attribute-Based Encryption (ABE) is a cryptographic primitive in which the identity of the user is defined through a set of elements called attributes. The main purpose of ABE schemes is to establish simplified means of decryption by describing a group of authorized users. Each user is able to decrypt messages that were encrypted over the defined authorized sets of attributes, thus eliminating the need of user communication regarding the decryption key. There are two forms of ABE scheme : ciphertext-policy ABE (CP-ABE) [1] and key-policy ABE (KP-ABE) [2]. In the CP-ABE scheme, the data encrypted can be accessed only by the users whose attributes satisfy the security policy. The authorization is included in the encryption, making the permission to the data an implicit action. In the KP-ABE scheme, the access policy is written in the user's key. Each key has an associated access structure which describes the type of ciphertext it can decrypt. The access structure is a tree whose leaves are all the elements from the attribute set and whose nodes are conjunctions and disjunctions.

This paper discusses only the KP-ABE schemes for general circuits starting by describing previous work. This construction appears as a necessity for a new efficient scheme. In *Key-policy Attribute-based Encryption for Boolean Circuits from Bilinear Maps* [3] by Țiplea and Drăgan, an approach using one bilinear map is presented and the paper itself represents the starting point of this work, thus making this new construction an improvement to the previous one. Moreover, another approach that helped with creating the new KP-ABE scheme for general circuits is [4]. Both contributions will be presented throughout the paper.

The paper is organized into eight sections. The next section contains theoretical notions which will be recalled throughout the presentation of the new KP-ABE scheme for general circuits and will provide a better understanding of the new construction. Before detailing the construction, the backtracking attack must be discussed pointing out why its existence is such an important problem that needs to be addressed and prevented in the general circuits extension of KP-ABE schemes. In the following four sections the contribution is described together with its security soundess, complexity and the new possible extension that can be applied over it. Finally, the eighth section concludes the paper.

2. PRELIMINARIES

A *multiplicative group* is a set G equipped with the binary operation $\cdot$ which satisfies the following properties:

- $\forall x, y \in G, x \cdot y \in G$;
- $\exists e \in G$ so that $\forall x \in G, e \cdot x = x \cdot e = x$;
- $\forall x \in G, \exists y \in G$ so that $x \cdot y = y \cdot x = e$;
- $\forall x, y, z \in G, (x \cdot y) \cdot z = x \cdot (y \cdot z)$.

A multiplicative group G is called *cyclic* if there is an element $g \in G$ which can generate all the elements of the group G by repeatedly multiplying it with itself. Such elements are called *generators*.

Bilinear maps. Let there be G_1, G_2 two multiplicative cyclic groups of the same order. A *bilinear map* from $G_1 \times G_1$ to G_2 it is a function $e: G_1 \times G_1 \to G_2$ where:

- $e(g^a, g^b) = e(g, g)^{ab}$, for $\forall g \in G_1, a, b \in \mathbb{Z}$;
- $e(g, g)$ is a generator of G_2, for $\forall g \in G_1$.

The *decisional Diffie-Hellman problem* (DDH) is an indistinguishability problem in which there are two computational indistinguishable instances, (g^a, g^b, g^{ab}) and (g^a, g^b, g^c), where g, a, b are elements defined as above and c is a randomly chosen element of $\mathbb{Z}_p$. The *decisional bilinear Diffie-Hellman problem* (DBDH) is the problem of distinguishing two elements $e(g, g)^{abc}$ and $e(g, g)^z$, where e is a bilinear map defined from $G_1 \times G_1$ to G_2, g is a generator of G_1 and a, b, c, z are randomly chosen elements of $\mathbb{Z}_p$.

Access structures. Let U be a set of elements called *attributes*. A set S of nonempty subsets of U is called *authorized access structure* if it is used to define the authorized user in a cryptographic system. Any other subset of U that does not belong to S is called *unauthorized* and it is an element of $\bar{S}$, the set of unauthorized users.

A *Boolean circuit* is a digital logic circuit composed of logic gates and wires. A Boolean circuit is called *monotone* if it contains only AND and OR gates. Depending on the structure of the circuit, the Boolean circuit can be restructured using the *De Morgan's laws*. In propositional logic and Boolean algebra, De Morgan's laws are a set of transformation rules which allow conjunctions and disjunctions to be rewritten using negation: $\overline{A \cap B} = \bar{A} \cup \bar{B}$ and $\overline{A \cup B} = \bar{A} \cap \bar{B}$.

Thus, an AND gate can be replaced with an OR gate and vice versa. These laws will be later used in the physical transformation of the general circuit in the new construction.

A *general circuit* is a circuit composed of AND, OR and NOT gates, each having their standard number of input wires, but whose output wires can vary in number.

Shamir's secret sharing scheme is a scheme for sharing secrets that uses polynomial interpolation for reconstructing the secret. Given k pair $(x_1, y_1), \dots, (x_k, y_k)$, where $x_i \neq x_j, 1 \leq i < j \leq k$, there is a unique polynomial function $P(x)$ with the degree equal to $k - 1$ so that $P(x_i) = y_i, 1 \leq i \leq k$. The shared secrets $I_1, \dots, I_k$ are chosen as $I_i = P(x_i), 1 \leq i \leq n$ where $x_1, \dots, x_n$ are pairwise distinct. Having the shares $\{I_i | i \in A\}$, for some group A with $|A| = k$, the secret can be obtained using Lagrange's interpolation formula as:

$$S = \sum\nolimits_{i \in A} (I_i \cdot \prod\nolimits_{j \in A / \{i\}} \frac{x_i}{x_j - x_i})$$

A KP-ABE scheme consists of four probabilistic polynomial-time (PPT) algorithms and one deterministic polynomial-time (DPT) algorithm:

- $Setup(\lambda, n)$ is a PPT algorithm that takes as input the security parameter λ, outputing a set of public parameters PP and a master key MSK for the n attributes constructed in this phase;
- $Enc(m, A, PP)$ is a PPT algorithm that encrypts the message m according to the set of attributes A and to the public parameters PP outputing the ciphertext E;
- $KeyGen(C, MSK)$ is a PPT algorithm that generates a decryption key D for the access structure C, given as a circuit, using the master key MSK;
- $Dec(E, D)$ is a DPT algorithm that decrypts the ciphertext E using the decryption key D, returning the message m or the special symbol $\perp$.

3. THE BACKTRACKING ATTACK

For a better understanding of the backtracking attack in KP-ABE schemes for general circuits, a short overview of the construction in [1] is presented:

- let there be a bilinear map $e: G_1 \times G_1 \rightarrow G_2$ with G_1 and G_2 two multiplicative cyclic groups of order p and g a generator of G_1;

- in order to encrypt a message m, the value of $e(g,g)^{ys}$ is computed and multiplied with m, where y is a randomly chosen value from the setup phase and s is a randomly chosen value from the encryption phase;

- each user that requests a decryption key triggers the key generation phase in which the value y is shared from the top of the access structure to the bottom of it. The generated values outputed from the secret sharing are later used to compute the decryption key

- in order to decrypt a ciphertext $me(g,g)^{ys}$ the user gives the generated decryption key which is used to reconstruct bottom-up the value $e(g,g)^{ys}$.

The backtracking attack is an exploitation of information leakage present due to the functionality of OR gate that appears only when there is a fanout greater than one next to this logic gate. For example, given an access structure C as in Fig. 1, the scheme is susceptible to the attack near the attribute 3 on level 0. When the decryption key is constructed, a pair of values will be generated by the OR gate from the output wire to the input wires. Let us suppose that the gate q_1 receives from the output wire the value x. In the functionality of the OR gate is stated that for each element received, two new elements will be created with the property that the new elements are equal to the one currently received, thus each wire obtaining an identical element. In this case, the gate q_1 sends to each input wire the same value x: $x_1 = x_2 = x$. These values are exponents for the bilinear map e and are used to generate $e(g,g)^{sx_1}$ and $e(g,g)^{sx_2}$.

Let there be a user with the set of attributes $\{2, 4\}$. The decryptor will learn the value $e(g,g)^{sx_1}$, being able to find the value of $e(g,g)^{sx_2}$ just by noticing that $e(g,g)^{sx_1} = e(g,g)^{sx_2}$.

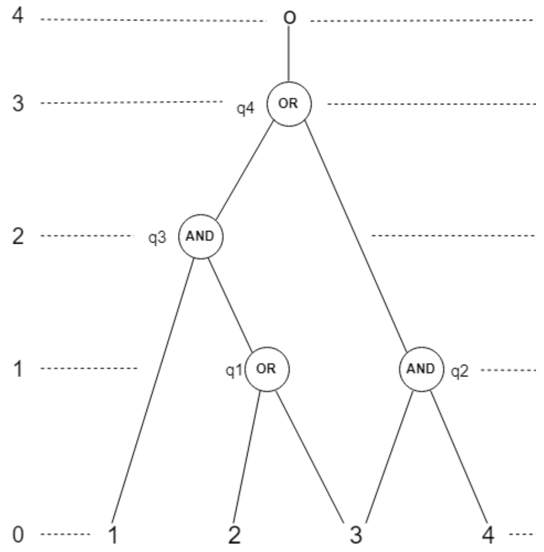

Fig. 1 General circuit C susceptible to backtracking attack

Previous works that use bilinear maps in the cryptographic scheme have prevented the backtracking attack through different techniques such as placing an additional logic gate called FO (fanout) to share the OR gate's secrets [3] and expanding the general circuit in order to force the size of the fanout to one [4]. Both contributions solve the backtracking attack problem although they come up with disadvantages.

In the new logic gate's case [3], the functionality of the FO gate requires random generation of two new values a and b so that $(a+b)\ mod\ p$ is equal to x, the value received from the output wires. This step is applied for every single element found in the output wires of the FO gate. All the values a are further shared throughout the access structure whereas all the values b are saved in a second storage and accessed only when the computation of the decryption key is needed. As it can be observed in the Fig. 2, the secret sharing algorithm outputs five main values and another two secondary values which are as important as the others in the construction of the decryption key. It is fairly easy to see that the scheme is rather costly when applied on complex general circuits. In the worst-case scenario the complexity of the decryption key can reach $nj + n + j^r$ elements, where n is the number of attributes, r is the number of FO gates with a maximum fanout size of j.

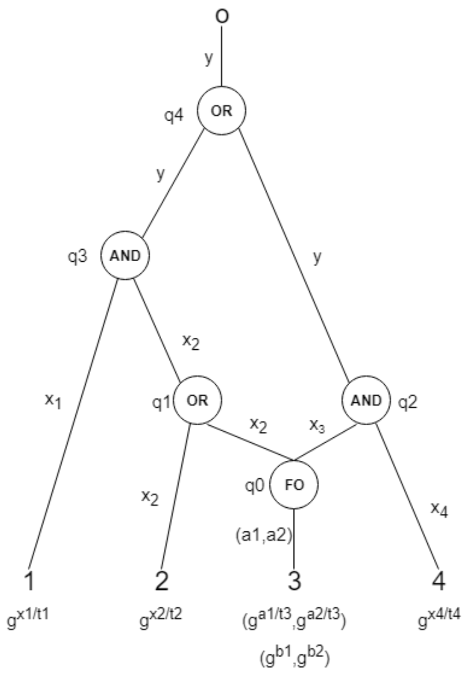

Fig. 2 Share algorithm applied on the general circuit C

In the second contribution [4], the backtracking attack is prevented by expanding the general circuits so that the size of the fanout is forced to one. The expansion algorithm starts from the bottom and goes up to the output wire of the circuit multiplying attributes, wires and gates when there is a fanout greater than one. At first, this techniques seems to be an easy and well needed solution which proved to be less costly than the previous one due to the nonexistence of FO gates. Taking a closer look at the scheme, it ignores the physical complexity of the new circuit. Thus, applying this solution over a complex general circuit will lead to a new massive access structure which is costly from a physical point of view. As it can be seen in the Fig. 3, the general circuit C has been expanded from a 3-attributes, 4-gates and 9-wires structure to a 6-attributes, 5-gates and 11-wires structure. As for the complexity of the decryption key, the scheme outputs in the worst-case scenario $n + j^r$ elements, where n is the number of attributes, r is the number of FO gates with a maximum fanout size of j.

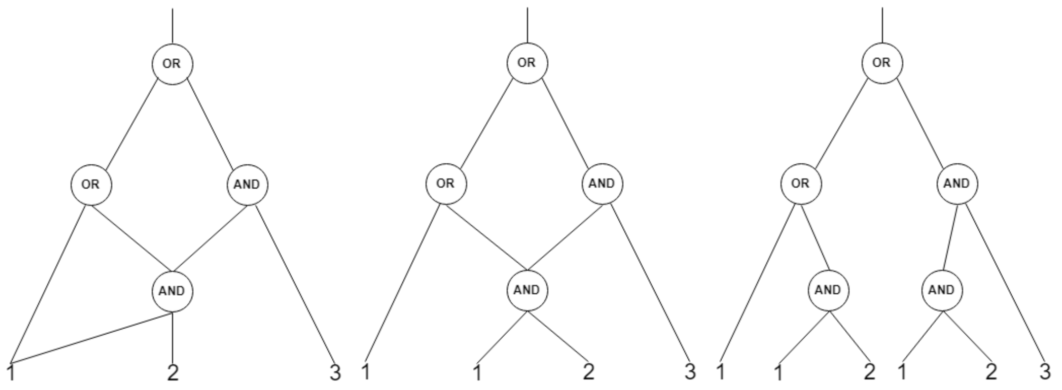

Fig. 3 Expansion of the general circuit C

Thus, a new solution which can solve these disadvantages is more than welcomed. For these reasons, a new scheme has been defined so that it keeps the number of shared secrets to a minimum and the size of the general circuit unmodified for an altogether better complexity.

4. A NEW CONSTRUCTION

As is can be observed, the biggest difficulty encountered throughout KP-ABE schemes for general circuits is the backtracking attack which appears only in OR gates near a fanout greater than one. Furthermore, due to its secure functionality, the AND gate is the most stable gate that the circuit has and will always withstand information leakage. In both papers [3] and [4], the discussion about the problem of the backtracking attack in general circuits surrounds the idea that the OR gates connected to a fanout greater than one is susceptible to the attack and describes solutions that maintain the problematic gate. In article [3], the prevention is made by securing the fanout through the use of FO gates. In article [4] the fanout is forced into having its size equal to one by expanding the general circuit. In either of these papers there are no modifications or improvements brought to the OR gate itself and no replacement with a more secure alternative is tried.

Thereby, a new construction is presented in which the OR gates are replaced with NAND gates whose functionality resembles the AND gate's, preventing in this way the backtracking attack and maintaining the size of the general circuit.

4.1 Circuit alteration

The access structure's modification is quite simple: all the OR gates are replaced with NAND gates, as seen in Fig. 4. The idea of altering the general circuit came to mind after seeing the De Morgan's laws of transformation. The functionality of the new gate is discussed in the next subsection.

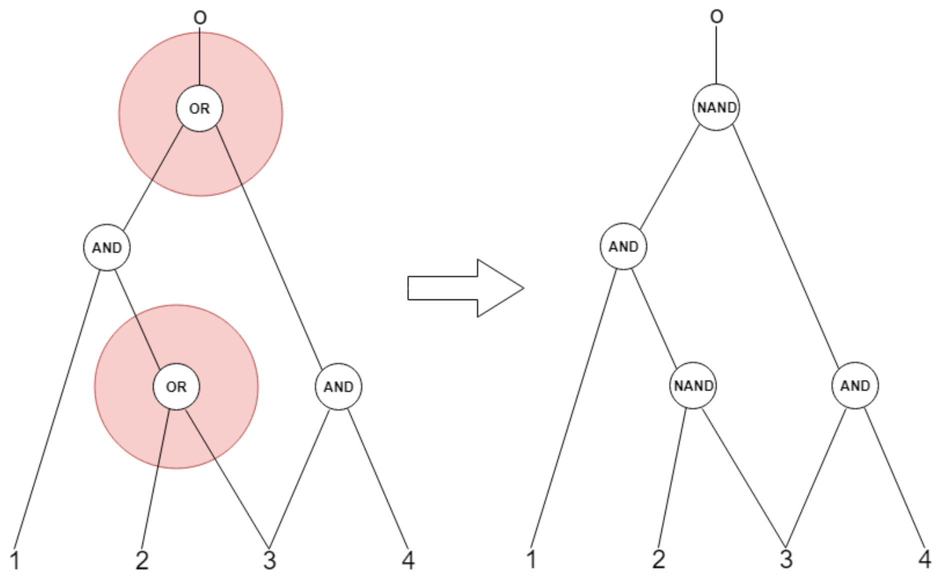

Fig. 4 Circuit alterations in the general circuit C

4.2 The new cryptographic system

The existence of a new logic gate brings alongside modifications in the KP-ABE scheme's algorithms. As stated above, the functionality of the NAND gate resembles the AND gate's. From all the existing algorithms, the secret sharing and the reconstruction algorithm are the most substantially modified. Therefore, the presentation will start with the four main algorithms of the KP-ABE scheme continuing afterwards with the previous two.

$Setup(\lambda, n)$:

- given the security parameter λ, a prime number p is chosen alongside two multiplicative cyclic group G_1 and G_2 of the same order p, a generator g of G_1 and a bilinear map $e: G_1 \times G_1 \rightarrow G_2$;
- let $U = \{1,2, \dots, n\}$ be the set of attributes;

- the element y is randomly chosen from $\mathbb{Z}_p$;
- for each $i \in U$, the element t_i is randomly chosen from $\mathbb{Z}_p$;
- the algorithm outputs the public parameters (PP) and the master key (MSK) defined as below:

$$PP = (p, G_1, G_2, g, e, n, Y = e(g,g)^y, (T_i = g^{t_i} | i \in U))$$
$$MSK = (y, t_1, \dots, t_n)$$

$Encrypt(m, A, PP)$:
- the element s is randomly chosen from $\mathbb{Z}_p$;
- the algorithm outputs the ciphertext E:

$$E = (A, E' = mY^s, (E_i = T_i = g^{t_i s}), g^s)$$

$KeyGen(C, MSK)$:
- the algorithm $Share(y, C)$ is called, saving the returned value in S;
- the algorithm outputs $D = (D(i) | i \in U)$, where

$$D(i) = \left(g^{\frac{S(i,j)}{t_i}} \middle| 1 \le j \le |S(i)| \right)$$

for each $i \in U$.

$Decrypt(E, D)$:
- the sequence V_A is computed as $V_A = (V_A(i) | i \in U)$, where

$$V_A(i,j) = e\big(E_i, D(i,j)\big) = e\left(g^{t_i s}, g^{\frac{S(i,j)}{t_i}} \right) = e(g,g)^{S(i,j)s}$$

for each $i \in A$ and $1 \le j \le |S(i)|$, otherwise $\perp$ if $i \in U - A$;
- the algorithm $Recon(C, V_A, g^s)$ is called, saving the returned value in R;
- the ciphertext is decrypted as following: $m = E'/R(o,1)$, where $R(o,1)$ represents the reconstructed value in the structure's output wire.

For a better understanding of the modified secret sharing and reconstructing algorithms, two figures of general circuits are introduced after each presentation.

$Share(y, C)$, where y is a value generated in the setup algorithm and C is the access structure represented as a general circuit, starts from the top of the circuit and ends when all gates have been visited:
- all the general circuit's gates are unmarked
- let S be a function that assigns to each wire of the circuit C a list of values from $\mathbb{Z}_p$;
- for the output wire, S assigns the list (y) as following: $S(o) = (y)$;
- if $(w_1, w_2, AND, W = (W_1, \dots, W_k))$ is an unmarked AND gate with two input wires w_1 and w_2, k output wires $W_1, \dots, W_k$ and $S(W_i) = L_i$, $1 \le i \le k$ then:
  - for each element $l \in L_i$, $x_l^1 \in \mathbb{Z}_p$ is randomly chosen and used to compute x_l^2 so that $l = \big(x_l^1 + x_l^2\big) \bmod p$, $1 \le i \le k$;
  - the sequences WL_1 and WL_2 are computed as following:
  $$WL_1 = \big((x_l^1 \big| for\ each\ l \in L_i) \mid 1 \le i \le k\big),$$
  $$WL_2 = \big((x_l^2 \big| for\ each\ l \in L_i) \mid 1 \le i \le k\big).$$
  - $S(w_1) = WL_1$ and $S(w_2) = WL_2$;
  - the gate is now marked.
- if $(w_1, w_2, NAND, W = (W_1, \dots, W_k))$ is an unmarked NAND gate with two input wires w_1 and w_2, k output wires $W_1, \dots, W_k$ and $S(W_i) = L_i$, $1 \le i \le k$ then:
  - for each element $l \in L_i$, $x_l^1 \in \mathbb{Z}_p$ is randomly chosen and used to compute x_l^2 so that $l = \big(-x_l^1 - x_l^2\big) \bmod p$, $1 \le i \le k$;
  - the sequences WL_1 and WL_2 are computed as following:
  $$WL_1 = \big((-x_l^1 \big| for\ each\ l \in L_i) \mid 1 \le i \le k\big),$$
  $$WL_2 = \big((-x_l^2 \big| for\ each\ l \in L_i) \mid 1 \le i \le k\big);$$
  - $S(w_1) = WL_1$ and $S(w_2) = WL_2$;
  - the gate is now marked.
- apply the same steps as above until all gates are marked.

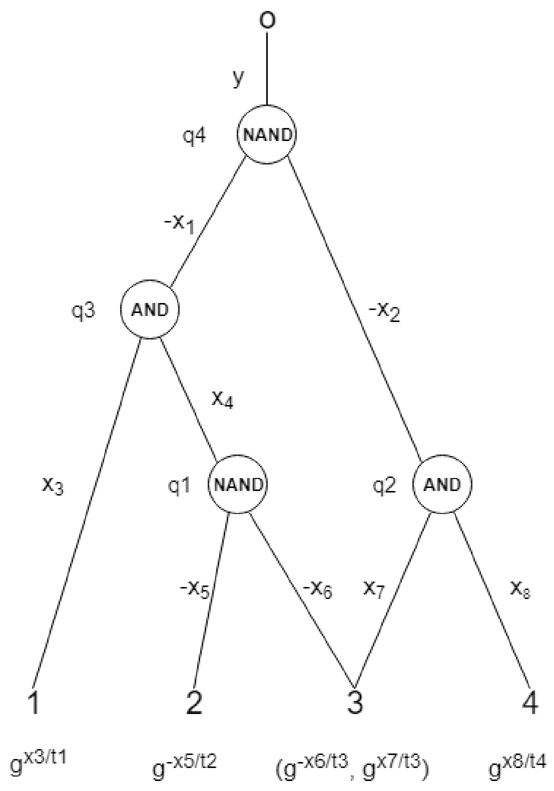

Fig. 5 Secret sharing algorithm applied on general circuit C

$Recon(C, V, g^S)$ is an algorithm that starts at the bottom of the access structure and going up reconstructing all shared secrets in the given circuit C from the sequence of values V as following:

- all the general circuit's gates are unmarked and let $S(i)$ be the list of values placed on the position i in $S(w)$, where w is a wire;

- each given sequence of values $V(i)$, where $i \in U$, is saved in $R(i)$;

- if $(w_1, w_2, AND, W = (W_1, \dots, W_k))$ is an unmarked AND gate with $R(w_1)$ and $R(w_2)$ already defined then mark the gate and compute:

$$R(W_i, j) = R(w_{1_i}, j) \cdot R(w_{2_i}, j), \text{ with } 1 \le i \le k \text{ and } 1 \le j \le |S_i(w_1)|;$$

- if $(w_1, w_2, NAND, W = (W_1, \dots, W_k))$ is an unmarked NAND gate with $R(w_1)$ and $R(w_2)$ already defined then mark the gate and compute:

$$R(W_i, j) = R(w_{1_i}, j) \cdot R(w_{2_i}, j), \text{ with } 1 \le i \le k \text{ and } 1 \le j \le |S_i(w_1)|;$$

- apply the same steps as above until all gates are marked.

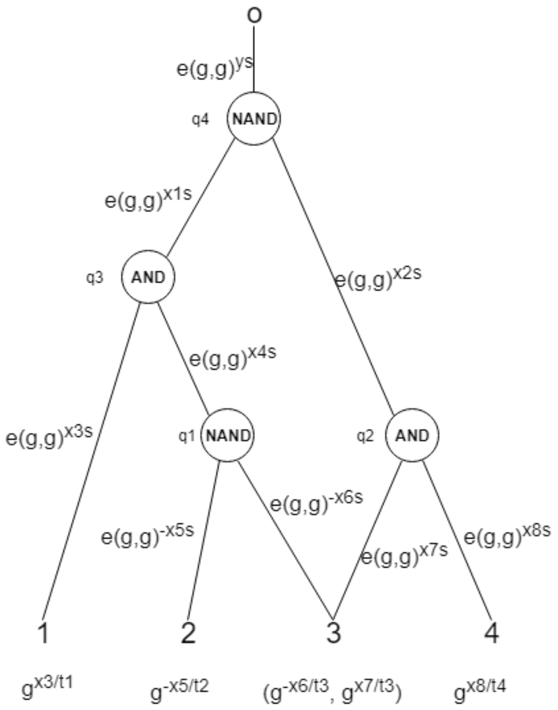

Fig. 6 Reconstruction algorithm applied on general circuit C

5. SECURITY

In this section, the security of the new construction is demonstrated by discussing the scheme's stability against the backtracking attack and by presenting the decisional bilinear Diffie-Hellman assumption through a cryptographic game using a PPT adversary and an oracle.

It is known that the backtracking attack problem is taken into consideration only when there are OR gates connected to a fanout with a size greater than one. In this model these gates are replaced with NAND ones eliminating the possibility of an attack due to their functionality. In order to prove that the new construction prevents the backtracking attack, the secret sharing algorithm will be detailed regarding NAND gates.

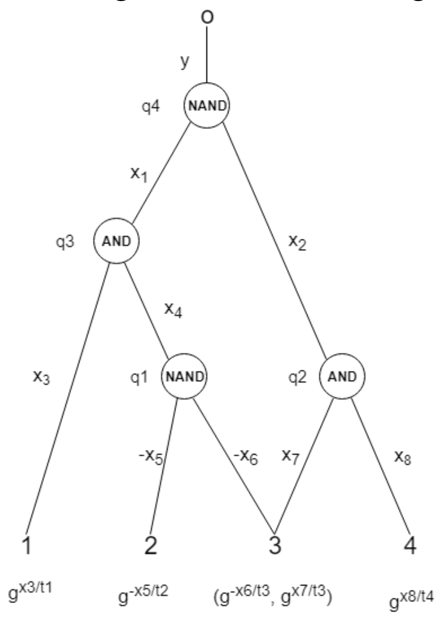

Fig. 7 Secret sharing algorithm on general circuit C

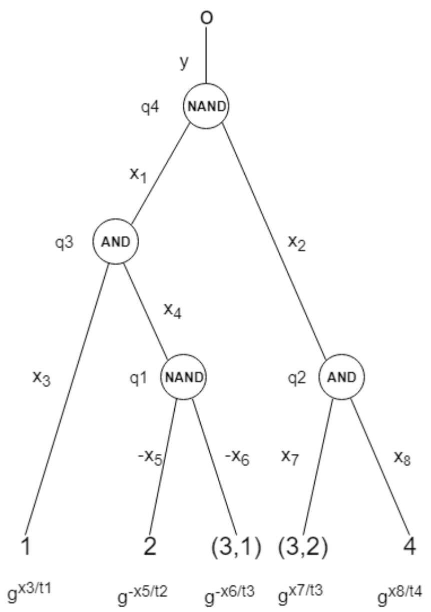

Fig. 8 Access tree for general circuit C

In the case of the general circuit C from Fig. 7, the functionality of the NAND gate imposes a random generation for $x_3 \in \mathbb{Z}_p$, whereas x_4 is defined so that $-x_3 - x_4 = x_2 \bmod p$. If the OR gates would have been kept then $x_3 = x_4 = x_2$, but in this construction the values are desired to be different from each other. Proving the security against backtracking attacks for NAND gates is equivalent to that of the AND gates. It is done by sharing secrets, the resulting values being the terms of the modular addition.

For a better understanding of the proof, the access tree of the general circuit C from Fig. 7 is built by multiplying the wires in the circuit based on the number of paths to the output. The paths are distinguished from each other through an identifier. Thus, for attribute 3 we have the separation of the wire in (3,1) and (3,2) because there are two paths to the output. It can be observed, both from the functionality of the NAND gate and from the access tree T_C that the backtracking attack cannot take place.

The second part of this section is dedicated to the security soundness of the new construction.

Let p be an odd prime number, G_1 and G_2 two multiplicative cyclic group of same order p, g a generator of G_1 and $e: G_1 \times G_1 \to G_2$ a bilinear map. The security game has a PPT adversary A and an oracle B. The oracle B receives an instance of the decisional bilinear Diffie-Hellman assumption (g^a, g^b, g^c, Z_v) with Z_v randomly chosen by B, where $Z_v \leftarrow \{Z_0, Z_1\}$, $Z_0 = e(g,g)^{abc}$ and $Z_1 = e(g,g)^z$, $a, b, c, z \in \mathbb{Z}_p$.

Init. Let M be a nonempty set of attributes chosen by the adversary A.

Setup. The oracle B chooses randomly a value $r_i \in \mathbb{Z}_p$ for each attribute $i \in U$, computes $Y = e(g^a, g^b) = e(g,g)^{ab}$ and $T_i = g^{t_i}$ for each attribute $i \in U$, where

$$t_i = \begin{cases} r_i, & i \in M \\ br_i, & otherwise \end{cases}$$

and outputs the public parameters

$$PP = (p, G_1, G_2, g, e, n, Y, (T_i | i \in U)).$$

First phase. The adversary gets access to the decryption key generation oracle for all queries in which the evaluation of the general circuit C for M is negative, meaning the set of attributes M is an unauthorized set. From the adversary's point of view, the secret sharing algorithm and the distribution of the decryption keys are the same as in the original scheme. The reconstruction using the $Recon$ algorithm must return $e(g,g)^{abc}$.

Once a request is received, the general circuit is converted into a tree access and the secret sharing process starts following the steps below:

- $S(o) = g^a$
- for $(w_1, w_2, AND, W = (W_1, \dots, W_k))$ with $S(W_j) = L_j$, where $1 \le j \le k$ then for each output wire W_j do:

1. if the evaluation of the output wire is positive for given M then for each $l \in L_j$ a randomly generated value $x_l^1 \in \mathbb{Z}_p$ is used to compute x_l^2 from $x_l^2 = (x_l^1 - l) \bmod p$. Two new lists are being defined:
$$L_1^j = (x_l^1 | for\ each\ l \in L_j)$$
$$L_2^j = (x_l^2 | for\ each\ l \in L_j).$$
$$S(w_1) = ((L_1^j) | 1 \le j \le k)$$
$$S(w_2) = ((L_2^j) | 1 \le j \le k).$$
2. if the evaluations of the output wire and the second input wire are negative but the evaluation of the first input wire is positive for given M then for each $l \in L_j$ a randomly generated value $x_l^1 \in \mathbb{Z}_p$ is used to compute $g^{x_l^2}$ from $g^{x_l^2} = l / g^{x_l^1}$. Two new lists are being defined:
$$L_1^j = (x_l^1 | for\ each\ l \in L_j)$$
$$L_2^j = (g^{x_l^2} | for\ each\ l \in L_j).$$
$$S(w_1) = ((L_1^j) | 1 \le j \le k)$$
$$S(w_2) = ((L_2^j) | 1 \le j \le k).$$

3. if the evaluations of the output wire and the first input wire are negative but the evaluation of the second input wire is positive for given M then for each $l \in L_j$ a randomly generated value $x_l^2 \in \mathbb{Z}_p$ is used to compute $g^{x_l^1}$ from $g^{x_l^1} = l/g^{x_l^2}$. Two new lists are being defined:
$$L_1^j = (g^{x_l^1} | for\ each\ l \in L_j)$$
$$L_2^j = (x_l^2 | for\ each\ l \in L_j).$$
$$S(w_1) = \left((L_1^j)\right| 1 \le j \le k)$$
$$S(w_2) = ((L_2^j)| 1 \le j \le k).$$
4. if all three evaluations are negative for given M then for each $l \in L_j$ a randomly generated value $x_l^1 \in \mathbb{Z}_p$ is used to compute $g^{x_l^2}$ from $g^{x_l^2} = l/g^{x_l^1}$. Two new lists are being defined:
$$L_1^j = (g^{x_l^1} | for\ each\ l \in L_j)$$
$$L_2^j = (g^{x_l^2} | for\ each\ l \in L_j).$$
$$S(w_1) = \left((L_1^j)\right| 1 \le j \le k)$$
$$S(w_2) = ((L_2^j)| 1 \le j \le k).$$

- for $(w_1, w_2, NAND, W = (W_1, \dots, W_k))$ with $S(W_j) = L_j$, where $1 \le j \le k$ then for each output wire W_j do:

1. if the evaluation of the output wire is positive for given M then for each $l \in L_j$ a randomly generated value $x_l^1 \in \mathbb{Z}_p$ is used to compute x_l^2 from $x_l^2 = (x_l^1 - l)\ mod\ p$. Two new lists are being defined:
$$L_1^j = (-x_l^1 | for\ each\ l \in L_j)$$
$$L_2^j = (-x_l^2 | for\ each\ l \in L_j).$$
$$S(w_1) = \left((L_1^j)\right| 1 \le j \le k)$$
$$S(w_2) = ((L_2^j)| 1 \le j \le k).$$
2. if the evaluations of the output wire and the second input wire are negative but the evaluation of the first input wire is positive for given M then for each $l \in L_j$ a randomly generated value $x_l^1 \in \mathbb{Z}_p$ is used to compute $g^{-x_l^2}$ from $g^{-x_l^2} = l/g^{-x_l^1}$. Two new lists are being defined:
$$L_1^j = (-x_l^1 | for\ each\ l \in L_j)$$
$$L_2^j = (g^{-x_l^2} | for\ each\ l \in L_j).$$
$$S(w_1) = \left((L_1^j)\right| 1 \le j \le k)$$
$$S(w_2) = ((L_2^j)| 1 \le j \le k).$$
3. if the evaluations of the output wire and the first input wire are negative but the evaluation of the second input wire is positive for given M then for each $l \in L_j$ a randomly generated value $x_l^2 \in \mathbb{Z}_p$ is used to compute $g^{-x_l^1}$ from $g^{-x_l^1} = l/g^{-x_l^2}$. Two new lists are being defined:
$$L_1^j = (g^{-x_l^1} | for\ each\ l \in L_j)$$
$$L_2^j = (-x_l^2 | for\ each\ l \in L_j).$$
$$S(w_1) = \left((L_1^j)\right| 1 \le j \le k)$$
$$S(w_2) = ((L_2^j)| 1 \le j \le k).$$
4. if all three evaluations are negative for given M then for each $l \in L_j$ a randomly generated value $x_l^1 \in \mathbb{Z}_p$ is used to compute $g^{-x_l^2}$ from $g^{-x_l^2} = l/g^{-x_l^1}$. Two new lists are being defined:
$$L_1^j = (g^{-x_l^1} | for\ each\ l \in L_j)$$
$$L_2^j = (g^{-x_l^2} | for\ each\ l \in L_j).$$
$$S(w_1) = \left((L_1^j)\right| 1 \le j \le k)$$
$$S(w_2) = ((L_2^j)| 1 \le j \le k).$$

The oracle B will give the adversary A the decryption key $D = (D(i)|i \in U)$ where

$$D(i) = \begin{cases} ((g^b)^{S_i(j)/r_i} \mid 1 \leq j \leq |S(i)|), i \in M \\ (S_i(j)^{\frac{1}{r_i}} \mid 1 \leq j \leq |S(i)|), otherwise \end{cases}, \forall i \in U.$$

The security game. The adversary A picks two messages of the same length m_0 and m_1 and sends them to the oracle B which encrypts the message m_u with Z_v where $Z_v \leftarrow \{Z_0 = e(g,g)^{abc}, Z_1 = e(g,g)^z\}$ and sending it back to the adversary. The set of parameters for the ciphertext is:

$$E = (M, E' = m_u Z_v, \{E_i = T_i^c = g^{cr_i}\}_{i \in M}).$$

If $v = 0$ then E is a valid encryption of the message m_u, otherwise E' is a random element from G_2.

Second phase. The adversary is granted access once again to the decryption key generation oracle under the same conditions as in the *first phase.*

Guessing. Let u' be the adversary's guess. If $u' = u$ then B returns $v' = 0$, otherwise $v' = 1$. The adversary's advantage is computed as following :

$$P(v' = v) - \frac{1}{2} = P(v' = 0|v = 0) \cdot P(v = 0) + P(v' = 1|v = 1) \cdot P(v = 1) - \frac{1}{2}$$

It is known that v is randomly chosen from $\{0, 1\}$ which means that $P(v = 0) = P(v = 1) = 1/2$. Moreover, it is observed that

$$P(v' = v|v = 0) = P(u' = u|v = 0) = \frac{1}{2} + \alpha$$
$$P(v' = v|v = 1) = P(u' \neq u|v = 1) = 1/2.$$

In conclusion, the advantage of B is equal to

$$P(v' = v) - \frac{1}{2} = \left(\frac{1}{2} + \alpha\right) \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} - \frac{1}{2} = \frac{1}{4} + \frac{\alpha}{2} + \frac{1}{4} - \frac{1}{2} = \frac{\alpha}{2}$$

, making the new construction a secure KP-ABE scheme for general circuits.

6. THE COMPLEXITY OF THE CONSTRUCTION

In this section, the complexity of the proposed model will be presented in comparison with the other two approaches described in section 3.

To show the complexity of the solution, the secret sharing algorithm must be detailed.

According to its functionality, the algorithm shares secrets through wires from output to input as following:

- The gate $(w_1, w_2, AND, W = (W_1, \ldots, W_k))$ sends a new value for each input wire w_1 and w_2 for each element found in $S(W)$;

- The gate $(w_1, w_2, NAND, W = (W_1, \ldots, W_k))$ also sends a new value for each input wire w_1 and w_2 for each element found in $S(W)$.

In conclusion, the secret sharing algorithm sends two new values for each element received from the output wires of the general circuit, thus the number of values depending strictly on the size of the circuit. It has been observed that in the worst-case scenario the number of elements grows exponentially based on the number of levels of the circuit, which was to be expected due to the algorithm's functionality. Therefore, we can discuss the complexity cases as follows:

I. The most favorable case is when a construction generates exactly n elements. The most simple and obvious example is when n can be written as 2^k with k being a natural number. In this situation all the wires receive strictly one value and thus on the attribute level there will be only n elements that build the decryption key;

II. The least favorable case is when the general circuit is structured on m levels in which all the wires communicate with the neighboring gates. The functionality of the secret sharing algorithm imposes on creating new values for each new input received through the output wires generating altogether a new list with these values for each single wire, thus creating double the number of values found on the previous level. The decryption key will have 2^m elements.

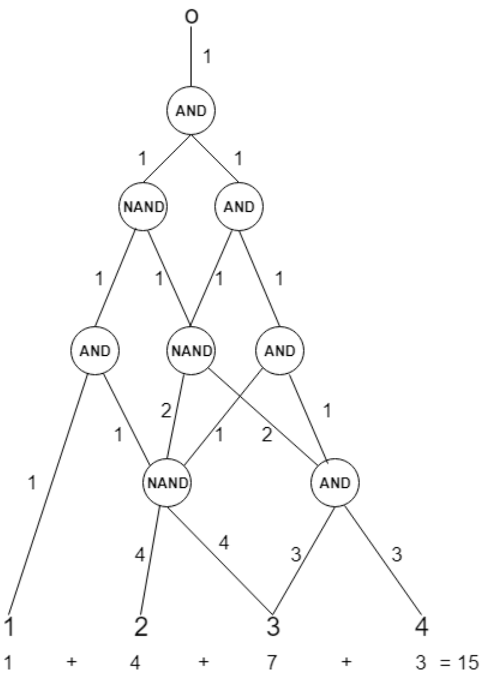

Fig. 9 Example of general circuit. Number of values for decryption key

Let n be the number of attributes and r the number of gates that have a fanout of size j. Hereinafter, the complexities of the other two KP-ABE schemes are compared to the proposed solution.

Case	Approach [3]	Approach [4]	Proposed solution
Most favorable case	$nj + n + r(j-1)$	$n + r(j-1)$	n
Least favorable case	$nj + n + j^r$	$n + j^r$	2^m

As it can be observed from the table, the number of elements from this cryptographic scheme depends on neither the size of the fanout, nor the number of fanouts. The proposed solution depends strictly on the number of levels of the general circuit.

For illustrating the differences of complexity, the solutions of the models [3] and [4] are applied on the general circuit C from Fig. 9.

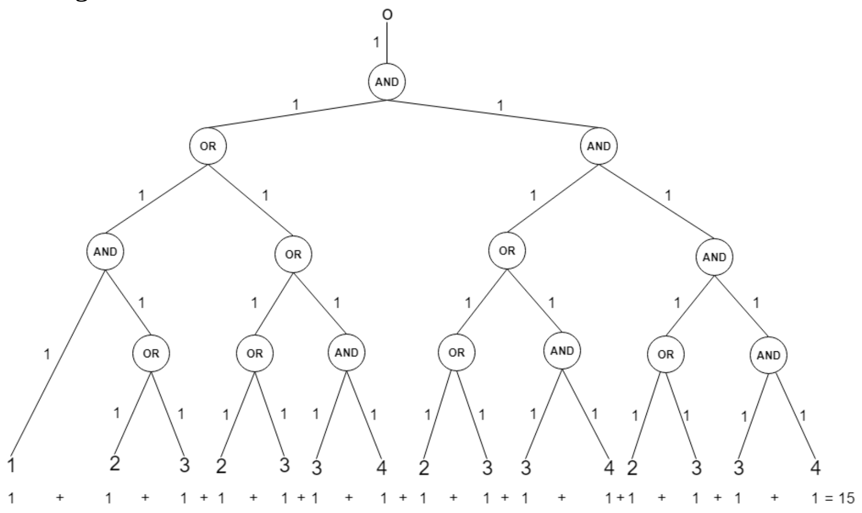

Fig. 10 Example of general circuit C. Number of values generated by the solution [4] for decryption key

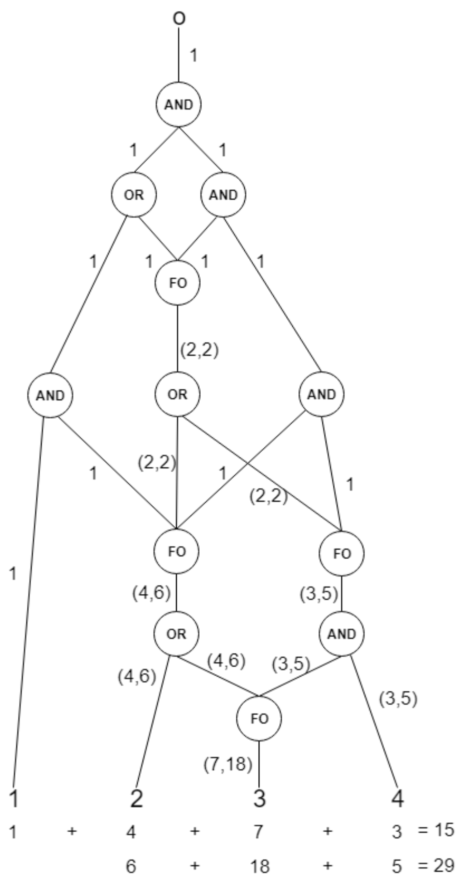

Fig. 11 Example of general circuit C. Number of values generated by the solution [3] for decryption key

As it can be observed from the images above, the cryptographic schemes defined in the papers [3] and [4] have their own drawbacks. In the case depicted in Fig. 11, the model generates in addition to the fifteen main elements, another twenty-nine secondary ones that are needed due to the functionality of the FO gate, while for the one in Fig. 10, no new variables are being computed but the dimension of the general circuit grows considerably.

7. EXTENSIONS

This solution cand be easily improved. As it can be observed, the proposed model is based on a general transformation in which no special constructions are being considered. An approach that could compress the general circuit, thus occupying less space by having a reduced number of gates, is constructing (k,n)-threshold gates from disjunctions of conjunctions such as depicted in the Fig. 12.

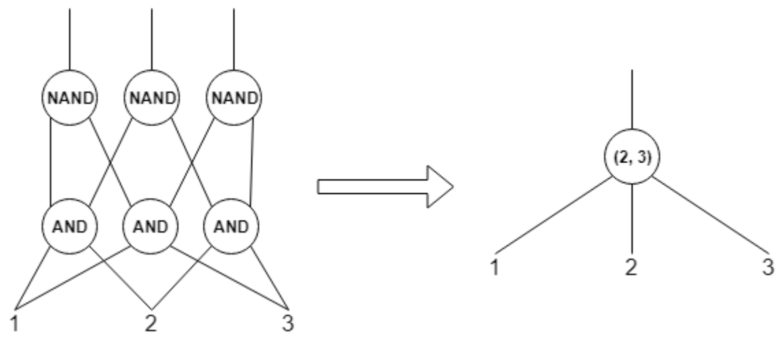

Fig. 12 Example of (2,3)-threshold

This solution could use Shamir’s scheme for sharing secrets. More details on this secret sharing scheme are discussed in the second section. For each secret introduced through the output wires of the (k,n) gate a new secret will be computed for each input wire based on the functionality of the Shamir’s secret sharing scheme. Intuitively, the number of elements of the decryption key will stagnate, though the physical space occupied by the general circuit will be reduced significantly.

From the computational efficiency’s point of view, it is known that some impediments might appear in the case of the modular inversion calculus, which is an operation that the polynomial interpolation is based on in the Shamir's scheme, but as a solution for this problem one could choose a prime number p represented as $2^k - 1$, where k is a natural number greater or equal to two, also known as a Mersenne number. The Mersenne numbers are used in the computation of modular inversions due to their form that allows the operations to be efficient.

8. CONCLUSIONS

The paper ”Key-Policy Attribute Based Encryption for General Circuits” manages to present a new viable KP-ABE approach for general circuits through maintaining the size of the access structure compared to the solution we saw in the paper [4] and reducing the number of elements of the decryption key by eliminating problematic gates and introducing a better, more secure alternative.

The efficiency of the proposed scheme depends strictly on the complexity of the general circuit, being a better solution compared to the approach in [3].

REFERENCES

1. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters, *Attribute-based encryption for fine-grained access control of encrypted data*, in ACM Conference on Computer and Communications Security, pages 89–98. ACM, 2006. Preprint on IACR ePrint 2006/309

2. John Bethencourt, Amit Sahai, and Brent Waters, *Ciphertext-policy attribute based encryption*, in IEEE Symposium on Security and Privacy, S&P 2007, pages 321–334. IEEE Computer Society, 2007

3. Ferucio Laurențiu Țiplea and Constantin Cătălin Drăgan, *Key-policy Attribute-based Encryption for Boolean Circuits from Bilinear Maps*, in BalkanCryptSec 2014, Istanbul, Turkey, 16-17 October, 2014, pages 175-193, LNCS 9024

4. Peng Hu and Haiying Gao, *A Key-Policy Attribute-based Encryption Scheme for General Circuit from Bilinear Maps*, in International Journal of Network Security, **Vol. 19**, No. 5, pages 704-710, September 2017

Received July, 10, 2019