Hierarchical and Shared Access Control

Arcangelo Castiglione, Alfredo De Santis, *Member, IEEE*, Barbara Masucci, Francesco Palmieri, Aniello Castiglione, *Member, IEEE*, Jin Li, and Xinyi Huang

***Abstract*— Access control ensures that only the authorized users of a system are allowed to access certain resources or tasks. Usually, according to their roles and responsibilities, users are organized in hierarchies formed by a certain number of disjoint classes. Such hierarchies are implemented by assigning a key to each class, so that the keys for descendant classes can be efficiently derived from classes higher in the hierarchy. However, pure hierarchical access may represent a limitation in many real-world cases. In fact, sometimes it is necessary to ensure access to a resource or task by considering both its directly responsible user and a group of users possessing certain credentials. In this paper, we first propose a novel model that generalizes the conventional hierarchical access control paradigm, by extending it to certain additional sets of qualified users. Afterward, we propose two constructions for hierarchical key assignment schemes in this new model, which are provably secure with respect to key indistinguishability. In particular, the former construction relies on both symmetric encryption and perfect secret sharing, whereas, the latter is based on public-key threshold broadcast encryption.**

***Index Terms*— Generalized access control, generalized access model, key assignment, provable security, shared key reconstruction, multiple access structures.**

I. Introduction

NOWADAYS the current network-centric world has given rise to several security concerns regarding access control management, which ensures that only authorized users are given access to certain resources or tasks. In particular, according to their respective roles and responsibilities, users are typically organized into *hierarchies* composed of several disjoint classes (*security classes*). A hierarchy is characterized by the fact that some users may have more access rights than others, according to a top-down inclusion paradigm following specific hierarchical dependencies. A user with access rights for a given class is granted access to objects stored in that class, as well as to all the descendant ones in the hierarchy. The problem of *key management* for such hierarchies consists of assigning a key to each class of the hierarchy, so that the keys for descendant classes can be efficiently obtained from users belonging to classes at a higher level in the hierarchy.

A. Hierarchical Key Assignment Schemes

A *hierarchical key assignment scheme (HKAS)* is a method for assigning an encryption key and some private information to each class in the system. The encryption key will be used by each class to protect its data by means of a symmetric cryptosystem, whereas, the private information will be used by each class to compute the keys assigned to all classes lower down in the hierarchy. The use of cryptographic techniques to address the key management problem in hierarchical structures was first considered by Akl and Taylor [1], who proposed a hierarchical key assignment scheme where each class is assigned a key that can be used, along with some public information generated by a trusted authority, to compute the key assigned to any class lower down in the hierarchy. Atallah et al. [2] first addressed the problem of formalizing security requirements for HKASs and proposed two different notions of security for such schemes: security against key recovery and with respect to key indistinguishability. In the former, an adversary is not able to compute a key which cannot be derived by the users he has corrupted; whereas, in the latter, the adversary is not able to distinguish the key from a random string of the same length. Atallah et al. also proposed the first provably-secure constructions based on pseudorandom functions and symmetric encryption schemes. Different constructions satisfying the above defined notions of security were subsequently proposed in [3]–[12]. In particular, De Santis et al. [4], [8] proposed two different constructions satisfying security with respect to key indistinguishability: the first one, which is based on symmetric encryption schemes, is simpler than the one proposed in [2], with it requiring a single computational assumption, and offering more efficient procedures for key derivation and key updates; the second one,

A. Castiglione, A. De Santis, B. Masucci, F. Palmieri, and A. Castiglione are with the Department of Computer Science, University of Salerno, Salerno 84084, Italy (e-mail: arcastiglione@unisa.it; ads@unisa.i; bmasucci@unisa.it; fpalmieri@unisa.it; castiglione@ieee.org).

J. Li is with the School of Computer Science, Guangzhou University, Guangzhou 510006, China (e-mail: lijin@gzhu.edu.cn).

X. Huang is with the Fujian Provincial Key Laboratory of Network Security and Cryptology, School of Mathematics and Computer Science, Fujian Normal University, Fuzhou 350117, China (e-mail: xyhuang81@gmail.com).

which is based on a public-key broadcast encryption scheme, allows to obtain a hierarchical key assignment scheme offering constant private information and public information that is linear in the number of classes. D'Arco et al. [6], [7] analyzed the Akl-Taylor scheme according to the definitions proposed in [2] and showed how to choose the public parameters in order to obtain instances of the scheme which are secure against key recovery under the RSA assumption. Moreover, they showed how to turn the Akl-Taylor scheme into a construction offering security with respect to key indistinguishability; however, such a scheme, is less efficient than the constructions proposed in [2], [4], and [8]. Freire et al. [13] proposed new security definitions for hierarchical key assignment schemes. Such definitions, called security against *strong key recovery* and security with respect to *strong key indistinguishability*, provide the adversary with additional compromise capability. Freire et al. showed how the notions of security against key recovery and against strong key recovery *are separated*. On the other hand, in [14] it has been proven that security with respect to strong key indistinguishability is *not stronger* than the one with respect to key indistinguishability, thus establishing the equivalence between such two security notions. A similar result has been shown in the unconditionally secure setting [15]. Finally, Ateniese et al. [3], [10] extended the model proposed in [2] to schemes satisfying additional time-dependent constraints and proposed two different constructions offering security with respect to key indistinguishability. Other constructions for time-dependent schemes, offering different trade-offs with respect to the amount of public and private information, as well as to the complexity of key derivation, were shown in [5], [9], [16], and [17].

B. Limitations of Hierarchical Key Assignment Schemes

In addition to conventional hierarchical access, sometimes it is necessary to provide some particular sets of users, having specific access credentials, with access to the key of a certain security class. This novel access control model finds a natural field of application even when there is the need to manage unusual, exceptional or emergency situations, which in general require special permissions. In particular, consider the case in which the trust is based on a single entity, let it be a person or an organization. Obviously, this may lead to abuses or violations by such entity, as in the Snowden event [18], where a great deal of confidential information held by the U.S. *National Security Agency* (*NSA*) was stolen. However, the NSA itself has defined in the past some strict guidelines for limiting such abuses, namely, the *Orange Book* [19] and *Two-Person Authorization* [20], [21], whose main goal was to prevent a single user from viewing top-secret documents. The concept upon which the guidelines are based is that, in general, somebody is less inclined to do something dishonest if someone else is watching. In addition, the two guidelines clearly state that the information within a system must be organized in a "*compartmental manner*", providing different levels of access and security to each compartment. In this case, a simple protection may be to use two or more "locks" to protect a given resource or activity, where each lock needs a different key, owned by a different person. Thus, two or more people are needed in order to grant the access to that resource or activity.

The Snowden event highlights the fact that the collaboration among several users and organizations is preferable for gaining the permission to carry out a given task or to access sensitive information. Such collaboration is needed so as to ensure that the requested permission has been granted through the acceptance and agreement among all the involved entities, thus preventing users from any kind of abuse. In general, the collaboration characterizes scenarios where more than one entity is required to achieve a specific authorization. More precisely, there are many real-world scenarios in which such a collaborative access is necessary, i.e., where a user might have a sort of "pre-authorization" for the access, but he may need to get the approval from someone else. For example, consider the healthcare environment, which typically consists of several professional profiles, such as doctors, nurses, etc.. In this environment, nurses may access a subset of stored patient's clinical data, while a doctor can usually access all the data. However, it is important to emphasize that the doctor and nurse must have the patient's consent to access clinical information. In addition, a nurse should not access all the information concerning a patient, unless she does not gain the permission from both patient and doctor. Moreover, if a doctor wants to access some clinical data without the explicit consent of the patient, he should be granted permission from several entities, e.g., hospital administration, medical committee, government authority, etc..

Again, the access to the workspace of a specific project branch could be granted either directly to the project manager or to a set of project team members. The same arguments apply to distributed cryptographic file systems [22]. A further real field of application lies in the collaborative access to logs concerning accesses and events, where the access can be achieved either by a single entity (e.g., a communications authority) or by more of them, which cooperate with each other. For example, the access might be allowed only if the judicial authority cooperates with a given service provider. Another concrete example arises from the military field, in which a decision can be taken by a single person with a specific rank, by a certain number of his subordinates or more generally, by a given number of people with certain credentials, which do not have the authority to decide on their own. Furthermore, consider a committee board composed of several members and a general chair. In this context, the chair might be away for personal reasons or could be in a situation which prevents him from making any decisions for a given action. Only one member of the board cannot independently take such a decision on behalf of the chair. However, the board members can collectively take such a decision on behalf of the chair, as long as their number is greater than or equal to a certain threshold.

All the aforementioned considerations and examples bring to light the fact that hierarchical and shared key assignment schemes are required in many cases. A *hierarchical and shared key assignment scheme* (HSKAS) should assign an encryption key and some private information to each class in the system in such a way that the private information of a group of qualified

TABLE I

FUNCTION ϕ OF THE DIRECTED MULTIGRAPH SHOWN IN FIGURE 1

e	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9
$\phi(e)$	(a,f)	(b,f)	(c,f)	(b,g)	(d,g)	(g,h)	(e,g)	(e,h)	(h,l)
e	e_{10}	e_{11}	e_{12}	e_{13}	e_{14}	e_{15}	e_{16}	e_{17}	e_{18}
$\phi(e)$	(f,i)	(g,i)	(g,l)	(a,f)	(e,f)	(c,f)	(d,f)	(b,h)	(e,h)

users is jointly required in order to compute the key assigned to a class lower down in the hierarchy.

C. Contributions

In this paper we first propose and formalize a novel access control model which prevents the abuse of permissions, defines alternative methods for gaining such permissions and allows the separation of duties. The model also enables collaboration among a set of users to gain specific permissions, defining the way in which such collaboration takes place. Furthermore, we formalize the notion of key indistinguishability regarding a new access model. Again, we propose two constructions which implement such a novel access control model. In particular, our first proposal, denoted as the *Shared Encryption Based Construction* (*SEBC*), uses as its basic building blocks a *symmetric encryption scheme* and a *perfect secret sharing scheme*, and offers security with respect to key indistinguishability. Our second proposal, denoted as the *Threshold Broadcast Encryption Based Construction (TBEBC)*, is based on a public-key *threshold broadcast encryption scheme* and provides security against key indistinguishability.

D. Organization

This paper is organized as follows. In Section II we formally define the model we propose, by focusing on its specific security properties. In Section III we provide our constructions for hierarchical and shared key assignment, along with the relative security proofs. Finally, in Section IV we draw some conclusions.

II. THE MODEL

Consider a set of users divided into a number of disjoint classes, called *security classes*. A security class can represent a person, a department, or a user group within an organization. According to their roles, competencies and responsibilities, security classes are often organized as a *hierarchy*.

There are several practical situations in which collaboration from users belonging to different security classes is needed, in order to access sensitive data belonging to a certain class lower down in the hierarchy. For example, a user could be part of multiple groups at the same time, and each of them could be associated to different access permissions. On the other hand, even the resource being accessed may require different access policies, according to those who access it. This problem can be solved by using a *hierarchical and shared access control*, where collaboration between classes is required to make sure that the access right to the private data held by a class v has been granted with the agreement of some particular users.

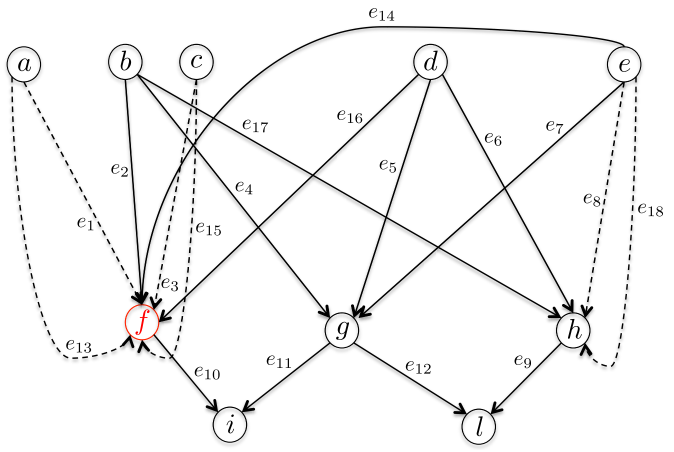

Fig. 1. Example of a directed multigraph characterizing our novel access control model.

All of these situations and usage scenarios can be modeled by means of a *directed multigraph*, namely, a directed graph that can have more than one edge between the same pair of vertices. The presence of an edge e connecting a class u to a class v means that class u is involved in a shared access control for class v's data. Moreover, since each class can obviously access the resources held by itself, the multigraph also contains "self-loops". However, from now on, for the sake of simplicity, we will not mention of such type of loops. Formally, a *directed multigraph* is a triple $G = (V, E, \phi)$, where V is a set of vertices, E is a set of edges, and ϕ is a function which associates each edge in E to its endpoints, that is: $\phi : E \to V \times V$. The edges e_1 and e_2 are said to be *multiple or parallel* if it holds that $\phi(e_1) = \phi(e_2)$. For example, consider the directed multigraph $G = (V, E, \phi)$ depicted in Figure 1, where $V = \{a, b, c, d, e, f, g, h, i, l\}$, $E = \{e_1, e_2, \cdots, e_{18}\}$ and ϕ is the function defined in Table I. Multiple edges are represented by dashed lines. In general, a subgraph $H = (V', E', \phi')$ of a multigraph $G = (V, E, \phi)$ is a multigraph whose underlying graph is a subgraph of that of G and its function ϕ' is dominated by ϕ, that is, the multiplicity of an edge in H does not exceed its multiplicity in G.

Why do we need multiple edges to model hierarchical and shared access control? The reason is that, in our new model, each class might be associated to different access structures, corresponding to different access rights/permissions. Given a directed multigraph $G = (V, E, \phi)$, for each class $v \in V$, let $\mathcal{I}_v \subset E$ be the set of edges ending in v. In this paper we consider the general situation where *more than one access structure* can be associated to each class in the multigraph. More precisely, for any $v \in V$, let $m_v \geq 1$ be an integer, let $\mathcal{P}_v^1, \ldots, \mathcal{P}_v^{m_v}$ be m_v subsets of $\mathcal{I}_v$, and let $\mathcal{A}_v^1, \ldots, \mathcal{A}_v^{m_v}$ be m_v access structures for class v on the sets $\mathcal{P}_v^1, \ldots, \mathcal{P}_v^{m_v}$, respectively. For each $v \in V$, let $\mathcal{A}_v = \{\mathcal{A}_v^1, \ldots, \mathcal{A}_v^{m_v}\}$ be the family of all access structures associated to class v and let $\mathcal{A}_G = \{\mathcal{A}_v\}_{v \in V}$. Notice that, if a hierarchical, but not shared, access control for a class v is required, we can simply associate to the class v a single access structure containing all the edges in $\mathcal{I}_v$, thus requiring no collaboration among classes in order to access v's data. In this case, two parallel edges make no difference, since they represent the same access right.

The proposed access model finds natural application especially in the so-called "*multi-domain environments*", namely, in all those environments in which there are different cooperating entities, each of them with different interests, responsibilities and tasks to perform. It is important to emphasize that a particular entity, depending on the context and the role which it assumes, may have different roles towards another given entity to which it intends to be granted access. Informally speaking, an entity can take on several tasks, and according to the assumed role it may have different access rights. On the other hand, a certain security class can provide the same entity with different access rights, according to the tasks performed by the latter in that particular context at that particular time. Clearly, at a given time an entity may need to take simultaneously all of its different tasks. In detail, an entity may assume one or more roles, each belonging to the characterization of a specific access structure, for a certain security class.

A. A Motivating Example

One of the most popular examples of multi-domain environments is obviously the one concerning healthcare. In particular, consider the multigraph shown in Figure 1, let f be the security class characterizing all the data related to a specific patient. It is easy to note that the set $\mathcal{I}_f$ is composed of seven elements, namely, $\mathcal{I}_f = \{e_1, e_2, e_3, e_{13}, e_{14}, e_{15}, e_{16}\}$. Without loss of generality, one of the possible characterizations for such set may be the following:

- $Entity_f^a$ (*Health director*) $= \{e_1, e_{13}\}$, where $Role_f^{e_1} =$ {Scientific Manager} and $Role_f^{e_{13}} =$ {Administrative Manager};
- $Entity_f^b = \{e_2\}$, where $Role_f^{e_2} =$ {Insurance Company};
- $Entity_f^c$ (*Doctor*) $= \{e_3, e_{15}\}$, where $Role_f^{e_3} =$ {Specialist} and $Role_f^{e_{15}} =$ {Common Practitioner};
- $Entity_f^d = \{e_{16}\}$, where $Role_f^{e_{16}} =$ {Patient's Family};
- $Entity_f^e = \{e_{14}\}$, where $Role_f^{e_{14}} =$ {Administrative Office}.

Therefore, starting from the $\mathcal{I}_f$ set, the following three access structures for the class f, that is to say, $\mathcal{A}_f^1$={{e_1, e_2, e_3}, {e_{13}, e_{14}},{e_{15}, e_{16}}}, $\mathcal{A}_f^2$={{e_{13}, e_{16}}, {e_2, e_{14}}, {e_1, e_3, e_{15}}} and $\mathcal{A}_f^3$={{e_2, e_{13}}, {e_{15}, e_{16}}, {e_1, e_3, e_{14}}}, might be characterized. For example, the first access structure, denoted as $\mathcal{A}_f^1$, could model the scenario in which the patient belonging to the class f is involved in a legal dispute for insurance issues. Instead, the access structure denoted as $\mathcal{A}_f^2$ can model the situation in which the patient intends to participate in the experimentation of a particular drug, therefore he needs to be subjected to some specific investigations and clinical evaluations. Finally, the access structure denoted as $\mathcal{A}_f^3$ can model the case in which the patient finds out to have a specific congenital disease, and for this reason needs to take out a new insurance policy on his health, consequently canceling the old one. Obviously, the three examples of access structure we provide for the class f, along with their relative characterizations, although concrete are extremely trivial. However, as it is easy to observe, the proposed model, because of its generality, is virtually able to represent any set of scenarios which may potentially occur in real life.

B. Hierarchical and Shared Key Assignment Schemes

Let $G = (V, E, \phi)$ be a directed multigraph and let $\mathcal{A}_G = \{\mathcal{A}_v\}_{v \in V}$ be the family of all access structures associated to the classes in V. For any $X \subseteq V$, we are going to define the set of classes A_X which can be accessed when classes in X collaborate together. Such a set will be constructed by using a *Breadth-First-Search (BFS)* on G, starting from the set X.

More precisely, starting from X, we will explore the multigraph G outgoing from X in all possible directions, adding classes one *layer* at a time, according to the access structures associated to the classes. First, notice that $X \in A_X$, since each class in X is authorized to access itself, with no need of cooperation. In particular, we denote by $A_X^0 = X$ the set of classes added at this step, also called *zero level of cooperation*. Then, we start from X and include all the classes $u \in V$ for which there exists a subset $Y \subseteq X$ such that the set E_Y of edges whose source classes are in Y belongs to *at least one* of the access structures associated to u. This is the *first layer of cooperation*, and the corresponding set of classes is denoted by A_X^1. We then include all the additional classes $u \in V$ for which there exists a subset $Y \subseteq A_X^1$ such that the set E_Y of edges whose source classes are in Y belongs to *at least one* of the access structures associated to u. This is the *second layer of cooperation*, and the corresponding set of classes is denoted by A_X^2. We continue in this way until there are no more layers to be explored, i.e., until the set $A_X^{diam(G)}$ has been constructed, where $diam(G)$ denotes the diameter of G. The set of classes A_X which can be accessed when classes in X collaborate together is naturally defined to be the union of the sets constructed in the different layers.

Formally, for each set of classes $X \subseteq V$ and for each level $j = 0, \ldots, diam(G)$, we define the set A_X^j corresponding to the set of classes which can be accessed by X at the j-th *level of cooperation*, as follows:

- $A_X^0 = X$;
- $A_X^j = \{v \in V : \exists Y \subseteq A_X^{j-1}$ s. t. $E_Y \in \mathcal{A}_v^i$, for some $i \in \{1, \ldots, m_v\}\}$.

The set of classes A_X which can be accessed when classes in X collaborate together is defined to be

$$A_X = \cup_{j=0}^{diam(G)} A_X^j.$$

Consider the multigraph depicted in Figure 1 and let $\mathcal{A}_f^1$={{e_1, e_2, e_3}, {e_{13}, e_{14}},{e_{15}, e_{16}}}, $\mathcal{A}_f^2$={{e_{13}, e_{16}}, {e_2, e_{14}}, {e_1, e_3, e_{15}}} and $\mathcal{A}_f^3$={{e_2, e_{13}}, {e_{15}, e_{16}}, {e_1, e_3, e_{14}}} be three access structures associated to class f.

Moreover, let $\mathcal{A}_g = \{\{e_4, e_5, e_7\}\}$, $\mathcal{A}_h = \{\{e_6, e_7\}, \{e_{17}, e_{18}\}\}$, $\mathcal{A}_i = \{\{e_{10}, e_{11}\}\}$, and $\mathcal{A}_l = \{\{e_9, e_{12}\}\}$ be the access structures associated to classes g, h, i, and l, respectively. Let $X = \{b, c, d, e\}$. It is easy to see that $A_X = \{b, c, d, e, f, g, h, i, l\}$. Indeed, $A_X^0 = \{b, c, d, e\}$, $A_X^1 = \{f, g, h\}$, and $A_X^2 = \{i, l\}$.

We are now ready to give a formal definition for *hierarchical and shared key assignment schemes*.

Notation: If $B(\cdot, \cdot, \ldots)$ is any probabilistic algorithm then $b \leftarrow B(x, y, \ldots)$ denotes the experiment of running B on inputs $x, y, \ldots$ and letting b be the outcome, the probability being over the coins of B. Similarly, if X is a set then $x \leftarrow X$ denotes the experiment of uniformly selecting an element from X and assigning x this value. If w is neither an algorithm nor a set then $x \leftarrow w$ is a simple assignment statement. A function $\epsilon : N \to R$ is *negligible* if, for every constant $c > 0$, there exists an integer n_c such that $\epsilon(n) < n^{-c}$ for all $n \geq n_c$.

Definition 1: A *hierarchical and shared key assignment scheme for* $(G, \mathcal{A}_G)$ is a pair (Gen, Der) of algorithms satisfying the following conditions:

1) The *information generation algorithm Gen* is probabilistic polynomial-time. It takes as inputs the security parameter 1^τ, a directed multigraph $G = (V, E, \phi)$ and the corresponding set of families of access structures $\mathcal{A}_G$, and produces as outputs:
   a) a private information s_u, for any class $u \in V$;
   b) a key $k_u \in \{0, 1\}^\tau$, for any class $u \in V$;
   c) a public information pub.

   We denote by (s, k, pub) the output of the algorithm Gen on inputs 1^τ, G and $\mathcal{A}_G$, where s and k denote the sequences of private information and of keys, respectively. Moreover, for any $X \subseteq V$, we denote by s_X the sequence of private information associated to the classes in X.
2) The *key derivation algorithm Der* is deterministic polynomial-time. It takes as inputs the security parameter 1^τ, a directed multigraph $G = (V, E, \phi)$ and the corresponding set of families of access structures $\mathcal{A}_G$, a set of classes $X \subseteq V$, the private information s_X held by classes in X, a class $u \in A_X$, and the public information pub, and produces as output the key k_u assigned to a class u.

   We require that for each class $u \in V$, each set of classes $X \subseteq V$, each sequence of private information s_X associated to classes in X, each class $u \in A_X$, each key k_u, each public information pub which can be computed by Gen on inputs 1^τ, G and $\mathcal{A}_G$, it holds that $Der(1^\tau, G, \mathcal{A}_G, X, s_X, u, pub) = k_u$.

C. Evaluation Criteria and Notions of Security

The efficiency of a hierarchical and shared key assignment scheme is evaluated according to several parameters, such as the amount of secret data that needs to be distributed to and stored by users, the amount of public data, the complexity of key derivation, and the resistance to collusive attacks. More precisely, for each class $u \in V$, the key k_u should be protected against any coalition of users which are not allowed to access such class, even when pooling together their private information.

Atallah et al. [2] first introduced two different security notions for hierarchical key assignment schemes where no cooperation between users is required: security with respect to *key indistinguishability* and security against *key recovery*. Security with respect to key indistinguishability formalizes the requirement that the adversary is not able to learn any information (even a single bit) about a key that it should not have access to, i.e., it is not able to distinguish it from a random string having the same length. On the other hand, security against key recovery corresponds to the weaker requirement that an adversary is not able to compute a key that it should not have access to. Examples of hierarchical key assignment schemes requiring no shared access control and satisfying the above requirements can be found in [3], [4], and [6]–[10]. In this paper we only consider security with respect to key indistinguishability and against polynomially bounded adversaries. Instead, two efficient constructions for the *unconditionally secure* setting may be found in [15], [23], and [24].

We consider a *static adversary* $\mathtt{STAT}_{u,X}$ that wants to attack a class $u \in V$ and which is able to corrupt a set of classes X such that $u \notin A_X$. We define an algorithm $Corrupt_u(s, X)$ which, starting from the private information s generated by the algorithm Gen, and a coalition of classes X such that $u \notin A_X$, extracts the sequence of private information s_X associated to the classes in X. Two experiments are considered. In the first one, the adversary is given the key k_u, whereas, in the second one, it is given a random string ρ having the same length as k_u. It is the adversary's job to determine whether the received challenge corresponds to k_u or to a random string. We require that the adversary will succeed with probability only negligibly different from 1/2.

*Definition 2 [*IND-ST*]:* Let $G = (V, E, \phi)$ be a directed multigraph and let $\mathcal{A}_G$ be the corresponding set of families of access structures, let (Gen, Der) be a hierarchical and shared key assignment scheme, and let $\mathtt{STAT}_{u,X}$ be a static adversary which attacks a class $u \in V$ and corrupts a set of classes X such that $u \notin A_X$. Consider the following two experiments:

```
Experiment $\mathbf{Exp}^{\mathtt{IND-1}}_{\mathtt{STAT}_{u,X}}(1^\tau, G, \mathcal{A}_G)$
    $(s, k, pub) \leftarrow Gen(1^\tau, G, \mathcal{A}_G)$
    $s_X \leftarrow Corrupt_u(s, X)$
    $d \leftarrow \mathtt{STAT}_{u,X}(1^\tau, G, \mathcal{A}_G, pub, s_X, k_u)$
    return $d$
Experiment $\mathbf{Exp}^{\mathtt{IND-0}}_{\mathtt{STAT}_{u,X}}(1^\tau, G, \mathcal{A}_G)$
    $(s, k, pub) \leftarrow Gen(1^\tau, G, \mathcal{A}_G)$
    $s_X \leftarrow Corrupt_u(s, X)$
    $\rho \leftarrow \{0, 1\}^\tau$
    $d \leftarrow \mathtt{STAT}_{u,X}(1^\tau, G, \mathcal{A}_G, pub, s_X, \rho)$
    return $d$
```

The advantage of $\mathtt{STAT}_{u,X}$ is defined as

$$\mathbf{Adv}^{\mathtt{IND}}_{\mathtt{STAT}_{u,X}}(1^\tau, G, \mathcal{A}_G) = |Pr[\mathbf{Exp}^{\mathtt{IND-1}}_{\mathtt{STAT}_{u,X}}(1^\tau, G, \mathcal{A}_G) = 1] - Pr[\mathbf{Exp}^{\mathtt{IND-0}}_{\mathtt{STAT}_{u,X}}(1^\tau, G, \mathcal{A}_G) = 1]|$$

The scheme is said to be *secure in the sense of* IND-ST if, for each directed multigraph $G = (V, E, \phi)$, each family of access structures $\mathcal{A}_G$, each class $u \in V$ and each set of classes X such that $u \notin A_X$, the function $\mathbf{Adv}^{\mathtt{IND}}_{\mathtt{STAT}_{u,X}}(1^\tau, G, \mathcal{A}_G)$ is negligible, for each static adversary $\mathtt{STAT}_{u,X}$ whose time complexity is polynomial in τ.

In Definition 2 we have considered a static adversary attacking a class. A different kind of adversary, the *adaptive* one, could also be considered. Such an adversary is first allowed to access all public information as well as all private information of a number of classes of its choice; afterwards, it chooses the class it wants to attack. However, following the lines of [10], it can be shown that security against adaptive adversaries is (polynomially) equivalent to security against static adversaries. Hence, in this paper we will only consider static adversaries.

III. Constructions

In this section we propose two different constructions for hierarchical and shared key assignment schemes. The former, denoted as the *Shared Encryption Based Construction (SEBC)*, is based on symmetric encryption and perfect secret sharing schemes, whereas, the latter, denoted as the *Threshold Broadcast Encryption Based Construction (TBEBC)*, is based on threshold broadcast encryption schemes. Both the proposed constructions are provably secure with respect to key indistinguishability.

A. A Construction Based on Symmetric Encryption

In the following we consider the problem of constructing a hierarchical and shared key assignment scheme by using as its basic building blocks a symmetric encryption scheme and a perfect secret sharing scheme. Before describing our construction, we first recall the definitions of symmetric encryption schemes and perfect secret sharing schemes.

1) Symmetric Encryption Schemes: A *symmetric encryption scheme* is a triple $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ of algorithms satisfying the following conditions:

1) The *key-generation algorithm* $\mathcal{K}$ is probabilistic polynomial-time. It takes as input the security parameter 1^τ and produces as output a string key.
2) The *encryption algorithm* $\mathcal{E}$ is probabilistic polynomial-time. It takes as inputs 1^τ, a string key produced by $\mathcal{K}(1^\tau)$, and a message $m \in \{0,1\}^*$, and produces as output the ciphertext y.
3) The *decryption algorithm* $\mathcal{D}$ is deterministic polynomial-time. It takes as inputs 1^τ, a string key produced by $\mathcal{K}(1^\tau)$, and a ciphertext y, and produces as output a message m. We require that for any string key which can be output by $\mathcal{K}(1^\tau)$, for any message $m \in \{0,1\}^*$, and for all y that can be output by $\mathcal{E}(1^\tau, key, m)$, we have that $\mathcal{D}(1^\tau, key, y) = m$.

In the following we define what we mean by a *secure* symmetric encryption scheme. We formalize security with respect to *plaintext indistinguishability*, which is an adaption of the notion of *polynomial security* as given in [25]. We consider an adversary $A = (A_1, A_2)$ running in two stages. In advance of the adversary's execution, a random key (key) is chosen and kept hidden from the adversary. During the first stage, the adversary A_1 outputs a triple $(x_0, x_1, state)$, where x_0 and x_1 are two messages of the same length, and $state$ is some state information which could be useful later. One message between x_0 and x_1 is chosen at random and encrypted to give the challenge ciphertext y. In the second stage, the adversary A_2 is given y and $state$ and has to determine whether y is the encryption of x_0 or x_1. Informally, the encryption scheme is said to be secure with respect to a *non-adaptive chosen plaintext attack*, denoted by `IND-P1-C0` in [26], if every polynomial-time adversary A, which has access to the encryption oracle only during the first stage of the attack and has never access to the decryption oracle, succeeds in determining whether y is the encryption of x_0 or x_1 with probability only negligibly different from $1/2$ (*random guess*).

In an *adaptive chosen plaintext attack* the adversary is also allowed to access the encryption oracle during the second stage of the attack. Notice that security with respect to such attack has been shown to be equivalent to the one with respect to a non-adaptive chosen plaintext attack in [26], thus in this paper we will only consider security with respect to `IND-P1-C0`.

Definition 3 [`IND-P1-C0`*]:* Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme and let τ be a security parameter. Let $A = (A_1, A_2)$ be an adversary which has access to the encryption oracle only during the first stage of the attack and has never access to the decryption oracle. Consider the following two experiments:

```
Experiment $\mathbf{Exp}^{\mathtt{IND-P1-C0-1}}_{\Pi,A}(1^\tau)$
    $key \leftarrow \mathcal{K}(1^\tau)$
    $(x_0, x_1, state) \leftarrow A_1^{\mathcal{E}_{key}(\cdot)}(1^\tau)$
    $y \leftarrow \mathcal{E}_{key}(x_1)$
    $d \leftarrow A_2(1^\tau, y, state)$
    return $d$
```

```
Experiment $\mathbf{Exp}^{\mathtt{IND-P1-C0-0}}_{\Pi,A}(1^\tau)$
    $key \leftarrow \mathcal{K}(1^\tau)$
    $(x_0, x_1, state) \leftarrow A_1^{\mathcal{E}_{key}(\cdot)}(1^\tau)$
    $y \leftarrow \mathcal{E}_{key}(x_0)$
    $d \leftarrow A_2(1^\tau, y, state)$
    return $d$
```

The advantage of A is defined as

$$\begin{aligned}\mathbf{Adv}^{\mathtt{IND-P1-C0}}_{\Pi,A}(1^\tau) = |&Pr[\mathbf{Exp}^{\mathtt{IND-P1-C0-1}}_{\Pi,A}(1^\tau) = 1] \\ &-Pr[\mathbf{Exp}^{\mathtt{IND-P1-C0-0}}_{\Pi,A}(1^\tau) = 1]|.\end{aligned}$$

The scheme is said to be *secure* in the sense of `IND-P1-C0` if the advantage function $\mathbf{Adv}^{\mathtt{IND-P1-C0}}_{\Pi,A}(1^\tau)$ is negligible, for any adversary A whose time complexity is polynomial in τ.

The XOR Construction for Symmetric Encryption Schemes: In order to construct an encryption scheme secure in the sense of `IND-P1-C0` we could use a *pseudorandom function family*, an important cryptographic primitive originally defined by Goldreich *et al.* [27]. Loosely speaking, a distribution of functions is pseudorandom if it satisfies the following requirements: 1) It is easy to sample a function according to the distribution and to evaluate it at a given point; 2) It is hard to tell apart a function sampled according to the distribution from a uniformly distributed function, given access to the function as a block-box. A first construction, based on pseudorandom generators, was proposed in [27]. It is well known that pseudorandom generators can be constructed from one-way functions [28], [29]. The two most efficient constructions were proposed by Naor and Reingold [30]. In particular, they showed a first construction, based on the hardness of factoring Blum integers, and a second one, based on the decisional version of the Diffie-Hellman assumption. In their constructions, the cost of evaluating such functions is comparable to two modular exponentiations.

Consider the following construction, called the *XOR construction* [31], of a symmetric encryption scheme $\Pi_{XOR,\mathcal{F}} = (\mathcal{K}_{XOR}, \mathcal{E}_{XOR}, \mathcal{D}_{XOR})$ which is based on a pseudorandom function family $\mathcal{F} : \{0,1\}^\tau \times \{0,1\}^\tau \rightarrow \{0,1\}^\tau$:

- The key generation algorithm $\mathcal{K}_{XOR}$ outputs a random τ-bit key ρ for the pseudorandom function family $\mathcal{F}$, thus specifying a function F_ρ of the family.
- The encryption algorithm $\mathcal{E}_{XOR}$ considers the message x to be encrypted as a sequence of τ-bits blocks $x = x_1 \cdots x_n$ (padding is done on the last block, if necessary), chooses a random string r of τ bits and computes, for $i = 1, \ldots, n$ the value $y_i = F_\rho(r+i) \oplus x_i$. The ciphertext is $r||y_1 \cdots y_n$, where $||$ denotes string concatenation.
- The decryption algorithm $\mathcal{D}_{XOR}$, on input a ciphertext z, parses it as $r||y_1 \cdots y_n$ and computes, for $i = 1, \ldots, n$ the value $x_i = F_\rho(r+i) \oplus y_i$. The corresponding plaintext is $x = x_1 \cdots x_n$.

The encryption scheme $\Pi_{XOR,\mathcal{F}}$ has been shown to be secure in the sense of `IND-P1-C0` (see [26], [31]), assuming that $\mathcal{F}$ is a pseudorandom function family. An efficient implementation of such a scheme could be obtained by using the HMAC [32] to realize the pseudorandom function family $\mathcal{F}$.

2) Perfect Secret Sharing Schemes: A *secret sharing scheme* $\Sigma = (Share, Recover)$ is a pair of algorithms run by a *dealer* and a set $\mathcal{P}$ of n participants. The $Share$ algorithm is executed by the dealer who, given a secret s, computes some shares $s_1, \ldots, s_n$ of such secret and gives each participant one share. The shares are computed in such a way that only qualified subsets of participants can reconstruct the value of s, by using the $Recover$ algorithm on input their shares, whereas any other subset of participants, non-qualified to know s, cannot determine anything about the value of the secret. Secret sharing schemes were introduced by Shamir [33] and Blakley [34] and have found applications in several areas of Data Security. Shamir and Blakley analyzed the case in which only subsets of participants of cardinality at least h, for a fixed integer $h \leq n$, can reconstruct the secret. These schemes are called (h,n)-threshold schemes. Subsequently, Ito et al. [35] and Benaloh and Leichter [36] described a more general method of secret sharing. They showed how to realize a secret sharing scheme for any *access structure*, where the access structure is the family of all the subsets of participants that are able to reconstruct the secret. The survey by Stinson [37] contains an unified description of results in the area of secret sharing schemes.

In this paper with a boldface capital letter, say $\mathbf{Y}$, we denote a random variable taking values on a set, denoted by the corresponding capital letter Y, according to some probability distribution $\{Pr_{\mathbf{Y}}(y)\}_{y \in Y}$. The values such a random variable can take are denoted by the corresponding lower case letter. Let S be the set of secrets, $\{Pr_{\mathbf{S}}(s)\}_{s \in S}$ be a probability distribution on S and let a secret sharing scheme for secrets in S be fixed. Assume for the rest of the paper that $Pr(\mathbf{S} = s) > 0$ for all $s \in S$. Let $\mathcal{P}$ be the set of participants, and for any $P \in \mathcal{P}$, let us denote by $Sh(P)$ the set of all possible shares given to participant P. Given a set of participants $X = \{P_{i_1}, \ldots, P_{i_r}\}$, where $i_1 < i_2 < \cdots < i_r$, let $Sh(X) = Sh(P_{i_1}) \times \cdots \times Sh(P_{i_r})$. Any secret sharing scheme for secrets in S and a probability distribution $\{Pr_{\mathbf{S}}(s)\}_{s \in S}$ naturally induce a probability distribution on $Sh(X)$, for any $X \subseteq \mathcal{P}$. We denote such probability distribution by $\{Pr_{\mathbf{X}}(x)\}_{x \in Sh(X)}$.

In terms of the probability distribution on the secret and on shares given to participants, we say that a secret sharing scheme $\Sigma = (Share, Recover)$ for the access structure $\mathcal{A}$ is *perfect* if the following two conditions hold:

1) *Any subset $X \subseteq \mathcal{P}$ of participants enabled to recover the secret can compute the secret.* Formally, if $X \in \mathcal{A}$, then, for all $x \in Sh(X)$ with $Pr(\mathbf{X} = x) > 0$, a unique secret $s \in S$ exists such that $Pr(\mathbf{S} = s | \mathbf{X} = x) = 1$.
2) *Any subset $X \subseteq \mathcal{P}$ of participants not enabled to recover the secret has no information about the secret.* Formally, if $X \notin \mathcal{A}$, then, for all $s \in S$ and for all $x \in Sh(X)$ with $Pr(\mathbf{X} = x) > 0$, it holds that $Pr(\mathbf{S} = s | \mathbf{X} = x) = Pr(\mathbf{S} = s)$.

In detail, Condition 1) means that the value of the shares held by participants in the qualified set X completely determines the secret $s \in S$. Instead, Condition 2) means that the probability that the secret is equal to s given that the shares held by participants in the non-qualified set X correspond to the sequence x, is equal to the a priori probability that the secret is s. Therefore, no amount of knowledge of shares of participants not qualified to reconstruct the secret enables a Bayesian opponent to modify an a priori guess regarding which the secret is.

It is well known that, in any perfect secret sharing scheme, the size of the share given to any participant is at least the size of the secret [38]. The sample space of shares given to any group of participants in a perfect secret sharing scheme, as a function of the size of the set of secrets has also been considered [39]. In particular, Blundo *et al.* [39] proved the following result, which will be useful later.

Remark 4: Let $\mathcal{A}$ be an access structure on the set of participants $\mathcal{P}$. In any perfect secret sharing scheme for $\mathcal{A}$ for any $X \notin \mathcal{A}$, it holds that $Pr(\mathbf{X} = x) = 1/|S|$, for any $x \in Sh(X)$.

Shamir's Threshold Schemes: In the following we recall the (h,n)-threshold scheme proposed by Shamir [33]. Let $q > n$ be a prime number, let $s \in \mathbb{Z}_q$ be the secret to be shared among the n participants and let $h \leq n$ be a fixed threshold. Let $x_1, \ldots, x_n$ be n distinct non-zero elements in $\mathbb{Z}_q$ known to all the parties (since q is a prime, then we can take $x_j = j$). To set up the scheme, the dealer constructs a random polynomial $a(x)$ of degree at most $h-1$, having coefficients in $\mathbb{Z}_q$, in which the constant term is the secret s. The share for participant P_i is the point (x_i, y_i) of the polynomial $a(x)$.

The correctness and privacy of Shamir's scheme derive from the *Lagrange's interpolation theorem*, which states that for any h distinct values $x_{i_1}, \ldots, x_{i_h}$ and any h values $y_{i_1}, \ldots, y_{i_h}$, there exists a unique polynomial $a'(x)$ of degree at most $h-1$ over $\mathbb{Z}_q$ such that $a'(x_{i_j}) = y_{i_j}$, for $j = 1, \ldots, h$. To see that Shamir's scheme is correct, notice that every set of participants $\{P_{i_1}, \ldots, P_{i_h}\}$ holds h points $s_{i_1}, \ldots, s_{i_h}$ of the polynomial $a(x)$, hence each set can reconstruct it using Lagrange's interpolation and compute $s = a(0)$. The set of participants computes

$$a'(x) = \sum_{\ell=1}^{h} s_{i_\ell} \prod_{1 \le j \le h, j \ne \ell} \frac{x_{i_j} - x}{x_{i_j} - x_{i_\ell}}.$$

Notice that $a'(x_{i_\ell}) = s_{i_\ell} = a(x_{i_\ell})$, for $\ell = 1, \dots, h$. That is to say, $a'(x)$ and $a(x)$ are polynomial of degree at most $h-1$ which agree on h points, thus, by the uniqueness in the interpolation theorem, they are equal, and, in particular, $a'(0) = a(0) = s$.

In fact, they know that $y_{i_j} = a(x_{i_j})$, for $1 \le j \le h$. Since $a(x)$ has degree at most $h-1$, $a(x)$ can be written as $a(x) = a_0 + a_1 x + \cdots + a_{h-1} x^{h-1}$, where $a_0, \dots, a_{h-1}$ are unknown elements in Z_q and $a_0 = s$ is the secret. Thus, the participants obtain a system of h linear equations in the h unknowns $a_0 \dots, a_{h-1}$. Such a system can be represented in matrix form as $Aa = y$, where the coefficient matrix A is a Vandermonde matrix, whose determinant can be computed as $det(A) = \prod_{1 \le j < t \le h} (x_{i_h} - x_{i_j}) \bmod q$. Since the x_i's are all distinct, then $det(A) \ne 0$ and it follows that the system has a unique solution over the field Z_q. Therefore, the h participants can reconstruct the whole polynomial $a(x)$ and compute the secret $s = a(0)$.

On the other hand, any $h-1$ participants have no information about the secret s. Proceeding as above, the group of participants obtain a system of $h-1$ equations in h unknowns. Suppose they hypothesize a value s' for the secret. Since the secret is $a(0) = a_0$, this will yield a further equation, and the coefficient matrix of the resulting system of h equations in h unknowns will again be a Vandermonde matrix. As before, there will be a unique solution. Hence, for every hypothesized value s' of the secret, there is a unique polynomial $a'(x)$ such that $y_{i_j} = a'(x_{i_j})$ for any $j = 1, \dots, h-1$ and such that $s' = a'(0)$. Hence, no value of the secret can be ruled out, and thus a group of $h-1$ participants obtain no information about the secret.

3) The Shared Encryption Based Construction: In the following we consider the problem of constructing a hierarchical and shared key assignment scheme by using as building blocks a symmetric encryption scheme and a perfect secret sharing scheme.

Rationale Behind the Construction: The idea behind our construction is similar to the one used in the *EBC* (*Encrypted Based Construction*) [8]. In the proposed construction, each class $v \in V$ is assigned a private information s_v, an encryption key k_v, and a public information π_v, which is the encryption of k_v using the private information s_v as a key. Moreover, for each class $v \in V$ and for each edge $e \in \mathcal{I}_v$, there is a public value p_e. If no shared access control for class v's data is needed (recall that in this case we can consider the trivial access structure $\mathcal{A}_v = \mathcal{I}_v$), the value p_e is computed as the encryption of the secret s_v, using the private information s_u as a key, where u and v are the endpoints of the edge e, i.e., $\phi(e) = (u, v)$. On the other hand, if a shared access control on class v's data is needed, we have to consider the m_v access structures $\mathcal{A}_v^1, \dots, \mathcal{A}_v^{m_v}$ associated to class v. The idea is to use a perfect secret sharing scheme for each $j = 1, \dots, m_v$, in order to compute the shares of the private

```
1:  procedure $Gen(1^\tau, G, \mathcal{A}_G)$
        ▷ Generation of sequences $s$ and $k$
2:      for each class $u \in V$ do
3:          $s_u \leftarrow \mathcal{K}(1^\tau)$, $k_u \leftarrow \{0,1\}^\tau$
4:      end for
        ▷ Generation of sequence $pub$
5:      for each class $u \in V$ do
6:          if $v$ requires access control according to $\mathcal{A}_v^1, \dots, \mathcal{A}_v^{m_v}$
    then
7:              for any $j = 1, \dots, m_v$ do
8:                  Use the $Share_v^j$ algorithm to generate the shares for the
                    secret $s_v$, according to $\mathcal{A}_v^j$
9:                  for each edge $e \in \mathcal{P}_v^j$ s.t. $\phi(e) = (u, v)$ do
                    ▷ Let $s_v^{j,u}$ be the corresponding share
10:                     Compute the public value $p_e \leftarrow \mathcal{E}_{s_u}(s_v^{j,u})$
11:                 end for
12:             end for
13:         else
14:             for each edge $e \in \mathcal{I}_v$, where $\phi(e) = (u, v)$ do
15:                 Compute the public value $p_e \leftarrow \mathcal{E}_{s_u}(s_v)$
16:             end for
17:         end if
18:         Compute the public value $\pi_v \leftarrow \mathcal{E}_{s_v}(k_v)$
19:     end for
20: end procedure
```

Fig. 2. *Gen* algorithm of the Shared Encryption Based Construction.

information s_v according to the access structure $\mathcal{A}_v^j$ on the set of edges $\mathcal{P}_v^j$. More precisely, let $e \in \mathcal{P}_v^j$ such that $\phi(e) = (u, v)$, and let $s_v^{j,u}$ be the share for the secret s_v associated to the edge e, according to the access structure $\mathcal{A}_v^j$. Such a share is encrypted with the private information s_u as a key and corresponds to the public value p_e associated to the edge e.

Given a class $v \in V$, any set of classes X such that $v \in A_X$ can obtain a set of shares for the secret s_v, decrypting some public values. Such shares allow for the computation of the secret s_v, which can then be used to decrypt the public value π_v, in order to get the key k_v. We will show that a static adversary attacking a class u and corrupting a set of classes X such that $u \notin A_X$, is not able to distinguish the key k_u from a random string of the same length unless it is able to break the underlying encryption scheme.

Let $G = (V, E, \phi)$ be a directed multigraph and let $\mathcal{A}_G$ be a family of access structures associated to classes in V. Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme, and, for any $v \in V$ and any $j = 1, \dots, m_v$, let $\Sigma_v^j = (Share_v^j, Recover_v^j)$ be a perfect secret sharing scheme for the access structure $\mathcal{A}_v^j$. The information generation algorithm *Gen* of the SEBC is shown in Figure 2, whereas, the relative key derivation algorithm *Der* is shown in Figure 3.

The SEBC associates a public value p_e to each edge $e \in E$, as well as a public value π_u to each class $u \in V$.

4) Analysis of the Scheme: In this section we show that the security property of the SEBC depends on the security properties of the underlying encryption scheme and of the perfect secret sharing scheme.

In particular, we show that if there exists an adversary able to break the security of the SECB in the sense of `IND-ST`, that is to say, which it is able to distinguish a value assigned by the SEBC from a randomly chosen one, then such an adversary can be used as a "*black-box*" to construct an

```
1:  procedure $Der(1^\tau, G, \mathcal{A}_G, X, s_X, u, pub)$
2:      if $u \in X$ then
3:          Extract $s_u$ from $s_X$ and the public value $\pi_u$ from $pub$
4:          $k_u \leftarrow \mathcal{D}_{s_u}(\pi_u)$
5:      else
            ▷ Let $1 \leq i \leq diam(G)$ be an index s.t. $u \in A_X^i$
6:          for each $\ell = 1, \ldots, i$ do
7:              for each class $v \in A_X^\ell$ do
                    ▷ Let $Y_v \subseteq A_X^{\ell-1}$ be s.t. $E_{Y_v} \in \mathcal{A}_v^j$ and $1 \leq j \leq m_v$
8:                  for each class $w \in Y_v$ do
9:                      Extract the public value $p_e$ from $pub$
                        ▷ Let $\phi(e) = (w, v)$
10:                     Compute the share $s_v^{j,w} \leftarrow \mathcal{D}_{s_w}(p_e)$
11:                 end for
                    ▷ On input the shares associated to edges in $E_{Y_v}$
12:                 Use $Recover_v^j$ algorithm to compute $s_v$
13:             end for
14:         end for
15:     end if
16:     Extract the public value $\pi_u$ from $pub$ and compute $k_u \leftarrow \mathcal{D}_{s_u}(\pi_u)$
17: end procedure
```

Fig. 3. *Der* algorithm of the Shared Encryption Based Construction.

adversary which breaks the underlying symmetric encryption scheme with respect of IND-P1-C0.

Our proof is essentially based on two well known concepts, referred to as *black-box reductions* and *hybrid arguments*. A black-box reduction is used to show that, given a cryptographic protocol P^f constructed from a cryptographic primitive f, if the protocol P^f can be broken somehow, then also the primitive f can be broken. Conversely, if f is believed to be secure, then also P^f must be secure. In general, all the security proofs for a protocol that do not use the internal structure of the primitives are called *black-box reductions*. On the other hand, the hybrid argument technique is used to argue that two *probability ensembles*, i.e., two sequences of probability distributions defined over the same probability space, are *computationally indistinguishable*. In this type of proof, one defines a sequence, constituted by a *polynomial number* (in the security parameter) of probability ensembles, also called the *hybrids*, where the extreme hybrids correspond to the two ensembles to be shown indistinguishable. In such a sequence, two adjacent hybrids differ only for a single value, corresponding to the application of a cryptographic primitive. Due to the total number of hybrids being polynomial, a non-negligible gap between the extreme hybrids translates into a non-negligible gap between a pair of adjacent hybrids and thus corresponds to breaking the security of the cryptographic primitive.

Theorem 5: If the encryption scheme $\Pi = (\mathcal{K}, \mathcal{D}, \mathcal{E})$ is secure in the sense of IND-P1-C0 and Σ is a perfect secret sharing scheme, then the SEBC is secure in the sense of IND-ST.

Proof: Let $G = (V, E, \phi)$ be a directed multigraph, let $u \in V$ and let $\texttt{STAT}_{u,X}$ be a static adversary which attacks class u and corrupts a set of classes $X \subset V$ such that $u \notin A_X$. Let $G_u = (V_u, E_u, \phi_u)$ be the subgraph of G induced by the set of vertices $V_u = \{v \in V :$ there is a path from v to u in $G\}$ and let $G_{u,X} = (V_{u,X}, E_{u,X}, \phi_{u,X})$ be the subgraph of G_u induced by the set of vertices $V_{u,X} = V_u \setminus X$, containing the classes in V_u which have not been corrupted by $\texttt{STAT}_{u,X}$. Without loss of generality, let $(u_1, \ldots, u_m)$, where $u_m \equiv u$, be any topological ordering of the vertices in $V_{u,X}$ and let $(e_1, \ldots, e_{h-1})$ be the sequence of edges in $E_{u,X}$ such that $\phi_{u,X}(e_i) = (u_a, u_b)$ precedes $\phi_{u,X}(e_j) = (u_c, u_d)$ if and only if either $a < c$ or $a = c$ and $b < d$. Moreover, let $\phi_{u,X}(e_h) = (u, u')$.

In order to prove the theorem, we need to show that the adversary's views in experiments $\mathbf{Exp}^{\texttt{IND-1}}_{\texttt{STAT}_{u,X}}$ and $\mathbf{Exp}^{\texttt{IND-0}}_{\texttt{STAT}_{u,X}}$ are indistinguishable. Notice that the only difference between $\mathbf{Exp}^{\texttt{IND-1}}_{\texttt{STAT}_{u,X}}$ and $\mathbf{Exp}^{\texttt{IND-0}}_{\texttt{STAT}_{u,X}}$ is the last input of $\texttt{STAT}_{u,X}$, which corresponds to the real key k_u in the former experiment and to a random value chosen in $\{0, 1\}^\tau$ in the latter. Thus, while in $\mathbf{Exp}^{\texttt{IND-1}}_{\texttt{STAT}_{u,X}}$ the public information is related to the last input of $\texttt{STAT}_{u,X}$, in $\mathbf{Exp}^{\texttt{IND-0}}_{\texttt{STAT}_{u,X}}$ it is completely independent on such a value. For ease of exposition, we define the following experiment:

```
Experiment $\mathbf{Exp}_{u,X}(1^\tau, G, \mathcal{A}_G)$
  $(s, k, pub) \leftarrow Gen(1^\tau, G, \mathcal{A}_G)$
  $s_X \leftarrow Corrupt_u(s, X)$
  $d \leftarrow \texttt{STAT}_{u,X}(1^\tau, G, \mathcal{A}_G, pub, s_X, \alpha_u)$
  return $d$
```

which corresponds either to $\mathbf{Exp}^{\texttt{IND-1}}_{\texttt{STAT}_{u,X}}$, if α_u is the real key k_u, or to $\mathbf{Exp}^{\texttt{IND-0}}_{\texttt{STAT}_{u,X}}$, if α_u is a random value in $\{0, 1\}^\tau$.

We will show that the adversary's view in the experiment $\mathbf{Exp}_{u,X}$ is indistinguishable from the adversary's view in an experiment $\mathbf{Exp}^*_{u,X}$, where the public information, at the same time, does not carry any information about the key k_u and is independent on the last input of $\texttt{STAT}_{u,X}$. More formally, the experiment $\mathbf{Exp}^*_{u,X}$ is defined as follows:

```
Experiment $\mathbf{Exp}^*_{u,X}(1^\tau, G, \mathcal{A}_G)$
  $(s, k, pub^*) \leftarrow Gen^*(1^\tau, G, \mathcal{A}_G)$
  $s_X \leftarrow Corrupt_u(s, X)$
  $d \leftarrow \texttt{STAT}_{u,X}(1^\tau, G, \mathcal{A}_G, pub^*, s_X, \alpha_u)$
  return $d$
```

In the algorithm Gen^* the public value π_u associated to class u is computed as the encryption $\mathcal{E}_{s_u}(\rho)$ of a random value $\rho \in \{0, 1\}^\tau$, rather than the encryption of the key k_u. Moreover, the public value associated to each edge e_i, where $\phi_{u,X}(e_i) = (u_a, u_b) \in E_{u,X}$ is computed as the encryption $\mathcal{E}_{s_{u_a}}(r_i)$ of a random value $r_i \in \{0, 1\}^\tau$, rather than the encryption $\mathcal{E}_{s_{u_a}}(s_{u_b}^{u_a})$ of the share $s_{u_b}^{u_a}$ for the private information s_{u_b}. Therefore, in such an experiment, all public information is independent on the value of the key k_u. Moreover, the distributions of the experiment $\mathbf{Exp}^*_{u,X}$ when $\texttt{STAT}_{u,X}$ is given as last input either the real key k_u or a random value in $\{0, 1\}^\tau$ are the same. In such an experiment, the key k_u is just a random value independent on the public and private information in the adversary's view.

Now, in order to show that $\mathbf{Exp}^{\texttt{IND-1}}_{\texttt{STAT}_{u,X}}$ and $\mathbf{Exp}^{\texttt{IND-0}}_{\texttt{STAT}_{u,X}}$ are indistinguishable, we only need to show that the adversary's views in experiments $\mathbf{Exp}_{u,X}$ and $\mathbf{Exp}^*_{u,X}$ are indistinguishable. This implies that $\mathbf{Exp}^{\texttt{IND-1}}_{\texttt{STAT}_{u,X}}$ and $\mathbf{Exp}^{\texttt{IND-0}}_{\texttt{STAT}_{u,X}}$ are both indistinguishable from the same experiment $\mathbf{Exp}^*_{u,X}$, which also means that they are indistinguishable from each other.

We construct a sequence of $h + 1$ experiments $\mathbf{Exp}^1_{u,X}, \ldots, \mathbf{Exp}^{h+1}_{u,X}$, all defined over the same probability space, where the first and the last experiments of the sequence correspond to $\mathbf{Exp}_{u,X}$ and $\mathbf{Exp}^*_{u,X}$. In each experiment, we modify the way in which the view of $\texttt{STAT}_{u,X}$ is computed, while maintaining the view's distributions indistinguishable among any two consecutive experiments. For any $q = 2, \ldots, h$, experiment $\mathbf{Exp}^q_{u,X}$ is defined as follows:

```
Experiment $\mathbf{Exp}^q_{u,X}(1^\tau, G, \mathcal{A}_G)$
  $(s, k, pub^q) \leftarrow Gen^q(1^\tau, G, \mathcal{A}_G)$
  $s_X \leftarrow Corrupt_u(s, X)$
  $d \leftarrow \texttt{STAT}_{u,X}(1^\tau, G, \mathcal{A}_G, pub^q, s_X, \alpha_u)$
  return $d$
```

The algorithm Gen^q used in $\mathbf{Exp}^q_{u,X}$ differs from Gen by the way in which part of the public information pub^q is computed. For any $i = 1, \ldots, q-1$, the public values associated to the edge e_i such that $\phi_{u,X}(e_i) = (u_a, u_b) \in E_{u,X}$ is computed as the encryption $\mathcal{E}_{s_{u_a}}(r_i)$ of a random value $r_i \in \{0,1\}^\tau$, instead of the encryption $\mathcal{E}_{s_{u_a}}(s^{u_a}_{u_b})$ of the share $s^{u_a}_{u_b}$.

In the following we show that, for any $q = 1, \ldots, h$, the adversary's view in the q-th experiment is indistinguishable from the adversary's view in the $(q+1)$-th one.

Assume by contradiction that there exists a polynomial-time distinguisher B_q, which is able to distinguish between the adversary $\texttt{STAT}_{u,X}$'s views in experiments $\mathbf{Exp}^q_{u,X}$ and $\mathbf{Exp}^{q+1}_{u,X}$ with non-negligible advantage. Notice that such views differ only for the way the public information associated to the edge e_q, such that $\phi_{u,X}(e_q) = (a,b)$, is computed. We show how to construct a polynomial-time adversary $A = (A_1, A_2)$ that uses B_q to break the security of the encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ in the sense of $\texttt{IND-P1-C0}$.

In particular, algorithm A_1, on input 1^τ, randomly chooses the key k_v for any class $v \in V$, as well as the private information for any class $v \in V \setminus \{a\}$.

If $I_a \neq \emptyset$, for any class $v \in I_a$, A_1 considers the m_a access structures $\mathcal{A}^1_a, \cdots, \mathcal{A}^{m_a}_a$ associated to the secret s_a and, for each of them, the set of edges characterizing that access structure. Then, A_1 computes the public value associated to each edge e such that $\phi_{u,X}(e) = (v, a)$ as the encryption of a random value chosen in $\{0,1\}^\tau$, using the private information s_v as a secret key. Afterwards, for any class $v \in V \setminus \{a\}$ such that $I_v \neq \emptyset$, A_1 considers the m_v access structures $\mathcal{A}^1_v, \cdots, \mathcal{A}^{m_v}_v$ associated to the secret s_v and, for each of them, it considers the set of edges characterizing such access structure. Subsequently, A_1 uses the algorithm $Share$, on input the secret information s_v, to compute the sequence of shares according to each access structure defined for the node v, which is characterized by the relative set of edges. Such shares will be used to compute part of the public information. More precisely, A_1 computes the public values associated to each edge e_r such that $\phi_{u,X}(e_r) = (v, z) \notin \{e_1, \ldots, e_q\}$ as the encryption of a random share relative to the secret information s_z by using s_v as a secret key. Notice that, in order to compute all public values associated to the outgoing edges of class a, except for the edge e_q such that $\phi_{u,X}(e_q) = (a,b)$, A_1 can make queries to the encryption oracle $\mathcal{E}_{s_a}(\cdot)$. Afterwards, A_1 computes the public values associated to each edge e_k such that $\phi_{u,X}(e_k) = (v,z) \in \{e_1, \ldots, e_{q-1}\}$ as the encryption of a random value chosen in $\{0,1\}^\tau$, using as a secret key the private information s_v. Again, A_1 computes the public values associated to all edges e_v such that $\phi_{u,X}(e_v) = (v, v')$, where $e_v \neq e_q$, as the encryption of the key k_v with the private information s_v. Finally, A_1 sets x_1 to be equal either to the key k_u, if $e_q = e_u$, where $\phi_{u,X}(e_u) = (u, u')$, or to a random share relative to the secret s_b, otherwise. Notice that, since Σ is a perfect secret sharing scheme, by Remark 4, such a share has the same distribution of a random value in $\{0,1\}^\tau$. The sequences s', k and pub' of all private information, keys, and public values constructed by A_1, along with the values x_0 and x_1, are saved in the state information $state$. Recall that the sequence s' contains the private information s_v assigned to all classes $v \in V \setminus \{a\}$. Similarly, the sequence pub' contains the public information associated to all edges in $E \setminus \{e_q\}$. Formally, the algorithm A_1 is defined as shown in Figure 4.

Let y be the challenge for the algorithm A, corresponding to the encryption of either x_0 or x_1 with the unknown key s_a. The algorithm A_2, on input 1^τ, y, and $state$, constructs the view for the distinguisher B_q as follows: it first extracts from s' the private information s_X held by corrupted users, through the algorithm $Corrupt_u(s', X)$. Then, it computes the public value associated to the edge e_q, not included in pub', in order to obtain the sequence pub. In particular, such a public value is set equal to the challenge y. Finally, A_2 outputs the same output as $B_q(1^\tau, G, \mathcal{A}_G, pub, s_X, x_1)$. More formally, A_2 works as shown in Figure 5.

Notice that if y corresponds to the encryption of x_1, then the random variable associated to the adversary's view is exactly the same as the one associated to the adversary view in experiment $\mathbf{Exp}^q_{u,X}$, whereas, if y corresponds to the encryption of x_0, it has the same distribution as the one associated to the adversary's view in experiment $\mathbf{Exp}^{q+1}_{u,X}$. Therefore, if the algorithm B_q is able to distinguish between such views with non negligible advantage, it follows that algorithm A is able to break the security of the encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ in the sense of $\texttt{IND-P1-C0}$. Contradiction.

Hence, for any $q = 1, \ldots, h$, the adversary's view in the q-th experiment is indistinguishable from the adversary's view in the $(q+1)$-th one. Therefore, the adversary's views in experiments $\mathbf{Exp}_{u,X}$ and $\mathbf{Exp}^*_{u,X}$ are indistinguishable. This concludes the proof. ■

B. A Construction Based on Threshold Broadcast Encryption

In this section we propose a construction for hierarchical and shared key assignment which uses as building block a *threshold broadcast encryption* scheme. We denote such a construction as the *Threshold Broadcast Encryption Based Construction (TBEBC)*. The TBEBC can be instantiated using the construction proposed by Daza et al. [40], which is secure under the *Decisional Bilinear Diffie-Hellman (DBDH)* assumption.

1) Threshold Broadcast Encryption: A broadcast encryption scheme allows a sender to broadcast an encrypted message to a set of users in such a way that only legitimate users can

```
procedure $A_1^{\mathcal{E}_{s_a}(\cdot)}(1^\tau)$
    $x_0, k_a \leftarrow \{0,1\}^\tau$
    for each $v \in V \setminus \{a\}$ do
        $s_v, k_v \leftarrow \{0,1\}^\tau$
    end for
    if $I_a \neq \emptyset$ then
        for each $v \in I_a$ do
        ▷ Consider the $m_a$ access structures $\mathcal{A}_a^1, \cdots, \mathcal{A}_a^{m_a}$ for $s_a$
            for $i = 1$ to $m_a$ do
                for each $e \in P_a^i$ do
                ▷ Let $\phi(e) = (v, a)$
                    $r_a \leftarrow \{0,1\}^\tau$
                    $p_{(v,a)} \leftarrow \mathcal{E}_{s_v}(r_a)$
                end for
            end for
        end for
    end if
    for each $v \in V \setminus \{a\}$ s.t. $I_v \neq \emptyset$ do
    ▷ Consider the $m_v$ access structures $\mathcal{A}_v^1, \cdots, \mathcal{A}_v^{m_v}$ for $s_v$
        for j = 1 to $m_v$ do
            $s_{v,\mathcal{A}_v^j} \leftarrow Share_v^j(s_v)$
            for each $e \in P_v^j$ s. t. $e \notin \{e_1, \ldots, e_q\}$ do
            ▷ Let $\phi(e) = (z, v)$
                $p(z, v) \leftarrow \mathcal{E}_{s_z}(s_v^{j,z})$
            end for
            if $q > 1$ then
                for each $e \in P_v^j$ s. t. $e \in \{e_1, \ldots, e_{q-1}\}$ do
                ▷ Let $\phi(e) = (z, v)$
                    $r_z \leftarrow \{0,1\}^\tau$
                    $p_{(z,v)} \leftarrow \mathcal{E}_{s_z}(r_z)$
                end for
            end if
        end for
    end for
    for each $(v, v') \in E$ do
        $\pi_v \leftarrow \mathcal{E}_{s_v}(k_v)$
    end for
    $pub' \leftarrow$ public values constructed above
    if $(a, b) = (u, u')$ then
        $x_1 \leftarrow k_u$
    else
    ▷ Consider the $m_b$ access structures $\mathcal{A}_b^1, \cdots, \mathcal{A}_b^{m_b}$ for $s_b$
        for $\ell = 1$ to $m_b$ do
            for each $e \in P_b^\ell$ do
                if $\phi(e) = (a, b)$ then
                    $x_1 \leftarrow s_b^{\ell,a}$
                end if
            end for
        end for
    end if
    $state \leftarrow (s', k, pub', x_0, x_1)$
    return $(x_0, x_1, state)$
end procedure
```

Fig. 4. First stage of the adversary A attacking the $SEBC$.

decrypt it. Broadcast encryption schemes can be either public-key or symmetric-key based. In the symmetric-key setting, only a trusted authority can broadcast data to the receivers. Conversely, in the public-key setting, a public key published by a trusted authority allows anybody to broadcast a message.

In a threshold public key broadcast encryption scheme (TBE) a message is encrypted and sent to a group of receivers, in such a way that the cooperation of at least t of them (where t is the threshold) is necessary in order to recover the original message. Such schemes have many applications in situations where one wants to avoid

```
procedure $A_2(1^\tau, y, state)$
    $state = (s', k, pub', x_0, x_1)$
    $s_X \leftarrow Corrupt_u(s', X)$
    ▷ Construction of the missing public values
    if $(a, b) = (u, u')$ then
        $\pi_u \leftarrow y$
    else
        $p_{(a,b)} \leftarrow y$
    end if
    $d \leftarrow B_q(1^\tau, G, \mathcal{A}_G, pub, s_X, x_1)$
    return $d$
end procedure
```

Fig. 5. Second stage of the adversary A attacking the $SEBC$.

that a single party has all the power/responsibility to protect or obtain some critical information. In those schemes, the sender of the message who wants to protect some information may want to decide who will be the designated receivers in an ad-hoc way, just before encrypting the message, and also decide the threshold of receivers which will be necessary to recover the information. More precisely, a TBE scheme has the following properties:

- There is no setup phase or predefined groups. Each potential receiver has his own pair of secret/public keys.
- The sender chooses the set of receivers $\mathcal{P}$ and the threshold t for the decryption. Then he encrypts the message by using the public keys of all the receivers in $\mathcal{P}$.
- A ciphertext corresponding to the pair ($\mathcal{P}$, t) can be decrypted only if at least t members of $\mathcal{P}$ cooperate by using their secret keys. Otherwise, it is computationally infeasible to obtain any information about the plaintext.

The next definition was proposed in [40].

Definition 6: A threshold broadcast encryption (TBE) scheme consists of five algorithms:

1) The randomized setup algorithm $TBE.Setup$ takes as input a security parameter 1^τ and outputs some public parameters $params$, which will be common to all the users of the system. We write $params \leftarrow TBE.Setup(1^\tau)$.
2) The randomized key generation algorithm $TBE.KG$ is run by each user i. It takes as input some public parameters $params$ and returns a pair (PK_i,SK_i) consisting of a public key and a matching secret key. We write $(PK_i, SK_i) \leftarrow TBE.KG(params)$.
3) The randomized encryption algorithm $TBE.Enc$ takes as input a set of public keys $\{PK_i\}_{i\in\mathcal{P}}$ corresponding to a set $\mathcal{P}$ of receivers, a threshold t satisfying $1 \le i \le n$, and a message m. The output is a ciphertext C. We write $C \leftarrow TBE.Enc(1^\tau, \{PK_i\}_{i\in\mathcal{P}}, t, m)$.
4) The (possibly randomized) partial decryption algorithm $TBE.PartDec$ takes as input a ciphertext C for the pair ($\mathcal{P}$,t) and a secret key SK_i of a receiver $i \in \mathcal{P}$. The output is a partial decryption value k_i or a special symbol $\perp$. We write $k_i \leftarrow TBE.PartDec(C, SK_i)$.
5) The deterministic final decryption algorithm $TBE.Dec$ takes as input a ciphertext C for the pair ($\mathcal{P}$,t) and t partial decryptions $\{k_i\}_{i\in A}$, corresponding to receivers in some subset $A \subset \mathcal{P}$. The output is a

message m or a special symbol $\bot$. We write $\widetilde{m} \leftarrow TBE.Dec(C, \{k_i\}_{i \in A}, A)$.

2) The Threshold Broadcast Encryption Based Construction: In the following we consider the problem of constructing a hierarchical and shared key assignment scheme by using as the building block a threshold broadcast encryption scheme.

Rationale Behind the Construction: The idea behind our construction, referred to in the following as the Threshold Broadcast Encryption Based Construction (TBEBC), is to compute the private and public information by using the threshold broadcast encryption scheme. More precisely, the public information associated to each security class v will contain a public key generated by a TBE, let PK_v be such a key. Given a class v with its relative access structures $\mathcal{A}_v^1, \ldots, \mathcal{A}_v^{m_v}$ and a qualified set $X \in \mathcal{A}_v^j$, for some $j \in \{1 \ldots m_v\}$, we denote with Q_X^v the set of classes having an outgoing edge in X. Furthermore, the public information will contain for each set Q_X^v a value given by the encryption of the secret key k_v, through the public keys of all the classes in Q_X^v. Subsequently, the public value relative to the set Q_X^v can be decrypted through the collaboration among all the classes that constitute this set; that is to say, through their private keys, the classes belonging to Q_X^v are allowed to compute the key k_v.

In detail, the generation algorithm of our TBEBC takes as inputs a security parameter 1^τ, a multigraph G, and the corresponding set of families of access structures $\mathcal{A}_G$. This algorithm generates a pair of public/private keys (PK_v, SK_v), for each class $v \in V$. Subsequently, for each class v, we assign secret information s_v, corresponding to the private key SK_v. Moreover, for each class v, the public key PK_v corresponds to the public information pub_v. In addition, for each class v, a secret key k_v is generated. Again, for each class v and for each set Q_X^v, k_v is encrypted through the public keys of the classes belonging to such a set. Finally, this encryption is assigned to the public information associated with the class v.

On the other hand, derivation algorithm of the TBEBC takes as inputs a security parameter 1^τ, a multigraph G, the corresponding set of families of access structures $\mathcal{A}_G$, the class u to be accessed, a set Q_X^u, the private information s_X^u associated to classes in Q_X^u and finally, all public information pub. This algorithm first extracts from pub the value C_X^u, relative to Q_X^u. Subsequently, it extracts from s_X^u the secret values associated to each class belonging to Q_X^u. Finally, through these values, the partial decryptions related to C_X^u are computed, for being used later in order to obtain the secret key k_u.

Formally, let $G = (V, E, \phi)$ be a directed multigraph and let $\mathcal{A}_G$ be a family of access structures associated to classes in V. Let $TBE = (TBE.Setup, TBE.KG, TBE.Enc, TBE.PartDec, TBE.Dec)$ a threshold broadcast encryption scheme. The information generation algorithm Gen of the TBEBC is shown in Figure 6, whereas, the relative key derivation algorithm Der is shown in Figure 7.

3) Analysis of the Scheme: In the following we show that the security property of the TBEBC depends on the security property of the underlying threshold broadcast encryption scheme. Before analyzing the security of the TBEBC, we first need to define what we mean by a *secure public-key threshold broadcast encryption scheme*. In general, in such schemes an adversary can corrupt different users in two possible ways: registering new public keys for such users, or obtaining the secret key matching with the public key of some previously honest users. The final goal of the adversary is to obtain some information about a message which has been encrypted for a pair $(\mathcal{P}^*, t^*)$, such that the number of corrupted players in $\mathcal{P}^*$ is less than t^*. For the ease of exposition, we consider the second kind of user corruption. More precisely, *indistinguishability* for TBE schemes is defined by considering the game shown in Figure 8, played by an adversary A_{atk} against a challenger [41].

In both phases of the attack, A_{atk} can access a corruption oracle $Corr$. In particular, A_{atk} submits to the oracle a user $i \in \mathcal{U}$ and receives as answer the relative secret key SK_i. Let $\mathcal{U}' \subset \mathcal{U}$ be the subset of users that A_{atk} has corrupted during the attack. Notice that $|\mathcal{P}^* \cap \mathcal{U}'| < t^*$ must hold,

```
1:  procedure $Gen(1^\tau, G, \mathcal{A}_G)$
2:      $params \leftarrow TBE.Setup(1^\tau)$
3:      for each class $v \in V$ do
4:          $(PK_v, SK_v) \leftarrow TBE.KG(params)$
5:          $k_v \leftarrow \{0,1\}^\tau$, $s_v \leftarrow SK_v$, $pub_v \leftarrow PK_v$
            ▷ Let $m_v \geq 1$ be an integer
            ▷ Let $P_v^1, \cdots, P_v^{m_v}$ be $m_v$ subsets of $I_v$
            ▷ Let $\mathcal{A}_i^1, \cdots, \mathcal{A}_v^{m_v}$ be the $m_v$ access structures for class $v$
              on the sets $P_v^1, \cdots, P_v^{m_v}$
6:          for $j = 1$ to $m_v$ do
            ▷ Let $Q_1^{v,j}, \cdots, Q_z^{v,j}$ be the collection of qualified sets char-
              acterizing the access structure $\mathcal{A}_v^j$
7:              for $k = 1$ to $z$ do
8:                  Let $card_k^{v,j}$ be the cardinality of the set $Q_k^{v,j}$
9:                  $C_k^{v,j} \leftarrow TBE.Enc(Q_k^{v,j}, \{PK_\ell\}_{\ell \in Q_k^{v,j}}, card_k^{v,j}, k_v)$
10:                 $pub_v \leftarrow C_k^{v,j}$
11:             end for
12:         end for
13:     end for
14: end procedure
```

Fig. 6. *Gen* algorithm of the Threshold Broadcast Encryption Based Construction.

```
1: procedure $Der(1^\tau, G, \mathcal{A}_G, Q_X^u, s_X^u, u, pub)$
2:     Extract from $pub$ the value $C_X^u$ associated to the class $u$
3:     for each $\ell \in Q_X^u$ do
4:         Extract from $s_X^u$ the secret value $s_\ell$ associated to the class $\ell$
5:         $Share_\ell \leftarrow TBE.PartDec(C_X^u, s_\ell)$
6:     end for
7:     $k_u \leftarrow TBE.Dec(C_X^u, \{Share_\ell\}_{\ell \in Q_X^u}, Q_X^u)$
8: end procedure
```

Fig. 7. *Der* algorithm of the Threshold Broadcast Encryption Based Construction.

```
$\mathcal{U} = \emptyset$
    $params \leftarrow TBE.Setup(1^\tau)$
    Each time $A_{atk}$ requires the creation of a new user $R_i$
        $(PK_i, SK_i) \leftarrow$ TBE.KG($params$)
        $\mathcal{U} \leftarrow \mathcal{U} \cup R_i$
    $(St, \mathcal{P}^*, t^*, m_0, m_1) \leftarrow A_{atk}^{Corr,\mathcal{O}_1}(\cdot)$(find, params, $\{PK_i\}_{i \in \mathcal{U}}$)
    $\beta \overset{r}{\leftarrow} \{0,1\}$
    $C^* \leftarrow$ TBE.Enc($\mathcal{P}^*$,$\{pk_i\}_{i \in \mathcal{P}^*}$,$t^*$,$m_\beta$)
    $\beta' \leftarrow A_{atk}^{Corr,\mathcal{O}_2}(\cdot)(guess, C^*, St)$
```

Fig. 8. Game played by an adversary A_{atk}.

otherwise, A_{atk} knows the secret key of at least t^* players in P^* and can decrypt C^* autonomously, obtaining m_β. In detail, depending on the considered type of attack, A_{atk} can also access a decryption oracle for ciphertexts of his choice. As an answer, A_{atk} receives all the information that would be broadcasted in a complete decryption process, that is, all the partial decryption values and the resulting plaintext. More precisely, if atk is a *Chosen Plaintext Attack (CPA)*, then the adversary cannot access the decryption oracle at all, i.e., $\mathcal{O}_1 = \mathcal{O}_2 = \epsilon$. If atk is a partial *Chosen Ciphertext Attack (CCA1)*, then $\mathcal{O}_1 = \text{TBE.PartDec}(\cdot) \cup \text{TBE.Dec}(\cdot)$ and $\mathcal{O}_2 = \epsilon$. Finally, if atk is a full *Chosen Ciphertext Attack (CCA2)*, then $\mathcal{O}_1 = \mathcal{O}_2 = \text{TBE.PartDec}(\cdot) \cup \text{TBE.Dec}(\cdot)$. Obviously, in the last case, A_{CCA2} is not allowed to query the oracle $\mathcal{O}_2$ with the challenge ciphertext C^*.
The advantage of A_{atk} is defined as:

$$\mathbf{Adv}(A_{atk}) = Pr[\beta' = \beta] - \frac{1}{2}.$$

A threshold broadcast encryption scheme is said to be ϵ-indistinguishable under atk attacks if $\mathbf{Adv}(A_{atk}) < \epsilon$ for any adversary A_{atk} running in polynomial time. Daza et al. [40] proposed a construction for threshold broadcast encryption schemes and showed it to be ϵ-indistinguishable under different kinds of attacks. Now we are ready to prove the next theorem.

Theorem 7: If the public-key threshold broadcast encryption scheme $TBE = (TBE.Setup, TBE.KG, TBE.Enc, TBE.PartDec, TBE.Dec)$ is ϵ-indistinguishable under atk attacks, then the $TBEBC$ is secure in the sense of `IND-ST`.

Proof: Assume by contradiction that the $TBEBC$ is not secure in the sense of `IND-ST`. Thus, there exists a multigraph $G = (V, E, \phi)$ in Γ and a class $u \in V$ for which there exists a polynomial-time adversary $\mathtt{STAT}_{u,X}$, whose advantage $\mathbf{Adv}^{\mathtt{IND}}_{\mathtt{STAT}_{u,X}}(1^\tau, G, \mathcal{A}_G)$ is non-negligible. We show how to construct a polynomial-time adversary A_{atk} that, by using $\mathtt{STAT}_{u,X}$, is able to break the ϵ-indistinguishability of the TBE scheme used as a building block in the $TBEBC$.

In particular, let a be the target class, let Q_a be a qualified set for a and finally, let $card_a = |Q_a|$. The first operation performed by the challenger is the generation of some parameters which will be used later on. The adversary A_{atk} chooses two messages, m_0 and m_1, both having the same length. For each node $v \in V$, A_{atk} asks the challenger for the creation of a new user v, along with the relative pair of public/private keys, denoted by PK_v and SK_v, respectively. It is important to remark that only public keys will be in the adversary's view. For each class $v \in V$, A_{atk} assigns the public key PK_v to the public information pub_v, besides assigning a secret key k_v to such a class. Again, the secret key k_a regarding the target class a is made to correspond to the message m_1. Subsequently, A_{atk} through its corruption oracle, corrupts a set X of users, such that a is not in A_X. Let s_X be the output of such a corruption, constituted by private keys associated to users in X. This output is then stored in the state variable St. After those steps, A_{atk} outputs some information through which it intends to be challenged. Later, the challenger computes an encryption C^* of a message chosen at random between m_0 and m_1, relative to the qualified set Q_a; let m_β be such a message. The encryption C^*, which represents the challenge for A_{atk}, is assigned to the public information for the node a, denoted by pub_a. By means of the aforementioned steps, A_{atk} simulated the full view for the distinguisher B_a, which is able to attack the security of $TBEBC$ in the sense of `IND-ST` with non-negligible advantage. More precisely, B_a is able to distinguish between the encryption of k_a from that of a random value. Finally, A_{atk} returns the same output as B_a, denoted by d. Formally, the adversary A_{atk} is defined as shown in Figure 9.

```
procedure $A_{atk}^{Corr,\mathcal{O}_1,\mathcal{O}_2}(1^\tau)$
    $params \leftarrow TBE.Setup(1^\tau)$
    $m_0, m_1 \leftarrow \{0,1\}^\tau$
    for each node $v \in V$ do
    ▷ $A_{atk}$ asks the challenger for the creation of a new user $v$
        $(PK_v, SK_v) \leftarrow TBE.KG(params)$
        $pub_v \leftarrow PK_v$, $k_v \leftarrow \{0,1\}^\tau$, $k_a \leftarrow m_1$
    end for
    ▷ Corruption of a set $X$ of users s.t. $a$ is not in $A_X$. Let $s_X$
      be the output of such a corruption
    $St \leftarrow s_X$
    $(St, Q_a, card_a, m_0, m_1) \leftarrow A_{atk}^{Corr,\mathcal{O}_1(\cdot)}(find, params, \{PK_v\}_{v \in V})$
    ▷ The challenger chooses at random a message between $m_0$
      and $m_1$ and provides the adversary with the encryption $C^*$
      of such a message
    $\beta \xleftarrow{r} \{0,1\}$
    $C^* \leftarrow TBE.Enc(Q_a, \{PK_j\}_{j \in Q_a}, card_a, m_\beta)$
    $pub_a \leftarrow C^*$
    ▷ Construction of the missing public values
    $d \leftarrow \mathcal{B}_a(1^\tau, G, A_G, pub, s_X, m_1)$
  $A_{atk}^{Corr,\mathcal{O}_2(\cdot)}(guess, C^*, St)$ returns the same $d$ as $B_a$
end procedure
```

Fig. 9. Functioning of the adversary $A_{atk}^{Corr,\mathcal{O}_1,\mathcal{O}_2}$ attacking the $TBEBC$.

Note that if the last input for $\mathtt{STAT}_{u,X}$ is equal to the key hidden into the public value C^*, then the random variable associated to $\mathtt{STAT}_{u,X}$'s view is exactly the same as in experiment $\mathbf{Exp}^{\mathtt{IND-1}}_{\mathtt{STAT}_{u,X}}$, whereas, if it is a random string, such a variable has the same distribution as the one associated to $\mathtt{STAT}_{u,X}$'s view in experiment $\mathbf{Exp}^{\mathtt{IND-0}}_{\mathtt{STAT}_{u,X}}$. Finally, A_{atk} outputs the same output as $\mathtt{STAT}_{u,X}(1^\tau, G, \mathcal{A}_G, pub, s_X, \alpha_u)$. Therefore, it holds that

$$\mathbf{Adv}(A_{atk}) = \mathbf{Adv}^{\mathtt{IND}}_{\mathtt{STAT}_{u,X}}(1^\tau, G, \mathcal{A}_G).$$

Since $\mathbf{Adv}^{\mathtt{IND}}_{\mathtt{STAT}_{u,X}}(1^\tau, G, \mathcal{A}_G)$ is non-negligible, it follows that the adversary A_{atk} is able to break the ϵ-indistinguishability of the threshold broadcast encryption scheme. Contradiction. ■

C. Performance Evaluation

The SEBC provides constant private information and public information linear in the number of the edges in the multigraph G. More precisely, the public information can be at most $(|E| + |V|)ck$, where k corresponds to the size of the secret key in this construction and c is a constant depending on the underlying symmetric encryption scheme. For instance, c is equal to 2 for the so called XOR construction in [31].

On the other hand, in the SEBC, for any $X \subseteq V$, the complexity of key derivation depends on the set A_X of classes that can be accessed when classes in X collaborate together. Such a set is constructed by using a

Breadth-First-Search (BFS) on G, starting from the set X. In detail, starting from X, we visit the multigraph G outgoing from X in all possible directions, adding classes one layer at a time, according to the access structures associated to the classes. Thus, besides the computational effort required by the BFS visit, the key derivation complexity in the SEBC is characterized by a number of decryptions which is equal to the number of classes corresponding to each layer of cooperation necessary to obtain the encryption key. Finally, in addition to the above number of decryptions, it is also necessary to employ the *Recover* algorithm of the perfect secret sharing scheme for reconstructing the secret key s_v.

Regarding the TBEBC, the number of public information associated to a security class v is given by $1 + z$, where z is the number of qualified sets that can access the class v. On the other hand, the only secret information assigned to each class v is given by the private key generated by the TBE scheme. Instead, concerning the complexity of key derivation, in the case of TBEBC the number of decryptions is equal to the number of classes belonging to a given qualified set for a class v.

IV. Conclusions

In this paper we have proposed an access control model with some innovative features. In particular, starting from the consideration that in some cases, besides the conventional hierarchical access, access should be granted to some qualified sets of users, the above model provides the user with the ability to prevent abuse of permissions, to define alternative access methods and to allow the separation of duties. Such a novel access model finds a natural field of application in several contexts. In general, our model characterizes any scenario where more than one entity is required to gain a specific authorization. In addition, such a model is useful in environments where it is necessary to address situations requiring special permissions.

Moreover, in this paper we provided the first formal definition of hierarchical and shared key assignment schemes. Again, we proposed an efficient construction for those schemes, denoted as *Shared Encryption Based Construction* (*SEBC*), which assigns to each class a single private information, whereas, the public information depends on the number of classes, as well as on the number of edges in the hierarchy. The security of the proposed construction relies on the ones of the underlying encryption and secret sharing schemes.

Finally, we proposed a construction based on public-key threshold broadcast encryption, denoted as *Threshold Broadcast Encryption Based Construction (TBEBC)*, which assigns to each class a single private information, whereas, the public information depends on the number of qualified sets which can access such a class. The security of the proposed construction relies on the one of the underlying threshold broadcast encryption scheme.

References

[1] S. G. Akl and P. D. Taylor, “Cryptographic solution to a problem of access control in a hierarchy,” *ACM Trans. Comput. Syst.*, vol. 1, no. 3, pp. 239–248, 1983. [Online]. Available: http://doi.acm.org/10.1145/357369.357372

[2] M. J. Atallah, M. Blanton, N. Fazio, and K. B. Frikken, “Dynamic and efficient key management for access hierarchies,” *ACM Trans. Inf. Syst. Secur.*, vol. 12, no. 3, 2009, Art. ID 18.

[3] A. De Santis, A. L. Ferrara, and B. Masucci, “Provably-secure time-bound hierarchical key assignment schemes,” in *Proc. ACM CCS*, 2006, pp. 288–297.

[4] A. De Santis, A. L. Ferrara, and B. Masucci, “Efficient provably-secure hierarchical key assignment schemes,” in *Proc. 32nd MFCS*, vol. 4708. 2007, pp. 371–382.

[5] M. J. Atallah, M. Blanton, and K. B. Frikken, “Incorporating temporal capabilities in existing key management schemes,” in *Proc. 12th Eur. Symp. Res. Comput. Secur. (ESORICS)*, Dresden, Germany, Sep. 2007, pp. 515–530. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-74835-9_34

[6] P. D’Arco, A. De Santis, A. L. Ferrara, and B. Masucci, “Security and tradeoffs of the Akl–Taylor scheme and its variants,” in *Proc. 34th MFCS*, vol. 5734. 2009, pp. 247–257.

[7] P. D’Arco, A. De Santis, A. L. Ferrara, and B. Masucci, “Variations on a theme by Akl and Taylor: Security and tradeoffs,” *Theoretical Comp. Sci.*, vol. 411, no. 1, pp. 213–227, 2010.

[8] A. De Santis, A. L. Ferrara, and B. Masucci, “Efficient provably-secure hierarchical key assignment schemes,” *Theoretical Comput. Sci.*, vol. 412, no. 41, pp. 5684–5699, 2011.

[9] A. De Santis, A. L. Ferrara, and B. Masucci, “New constructions for provably-secure time-bound hierarchical key assignment schemes,” *Theoretical Comput. Sci.*, vol. 407, nos. 1–3, pp. 213–230, 2008.

[10] G. Ateniese, A. De Santis, A. L. Ferrara, and B. Masucci, “Provably-secure time-bound hierarchical key assignment schemes,” *J. Cryptol.*, vol. 25, no. 2, pp. 243–270, 2012.

[11] E. S. V. Freire and K. G. Paterson, “Provably secure key assignment schemes from factoring,” in *Proc. 16th Australasian Conf. Inf. Secur. Privacy (ACISP)*, Melbourne, Vic., Australia, Jul. 2011, pp. 292–309. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-22497-3_19

[12] Y. Zhang, J. Chen, R. Du, L. Deng, Y. Xiang, and Q. Zhou, “FEACS: A flexible and efficient access control scheme for cloud computing,” in *Proc. IEEE 13th Int. Conf. Trust, Secur., Privacy Comput. Commun. (TrustCom)*, Beijing, China, Sep. 2014, pp. 310–319. [Online]. Available: http://dx.doi.org/10.1109/TrustCom.2014.42

[13] E. S. V. Freire, K. G. Paterson, and B. Poettering, “Simple, efficient and strongly KI-secure hierarchical key assignment schemes,” in *Topics in Cryptology* (Lecture Notes in Computer Science), vol. 7779. E. Dawson, Ed. San Francisco, CA, USA: Springer, Feb./Mar. 2013, pp. 101–114. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-36095-4_7

[14] A. Castiglione, A. De Santis, and B. Masucci, “Key indistinguishability vs. strong key indistinguishability for hierarchical key assignment schemes,” *IEEE Trans. Dependable Secure Comput.*, to be published. [Online]. Available: http://dx.doi.org/10.1109/TDSC.2015.2413415

[15] M. Cafaro, R. Civino, and B. Masucci, “On the equivalence of two security notions for hierarchical key assignment schemes in the unconditional setting,” *IEEE Trans. Dependable Secure Comput.*, vol. 12, no. 4, pp. 485–490, Jul./Aug. 2015. [Online]. Available: http://dx.doi.org/10.1109/TDSC.2014.2355841

[16] A. De Santis, A. L. Ferrara, and B. Masucci, “New constructions for provably-secure time-bound hierarchical key assignment schemes,” in *Proc. 12th ACM Symp. Access Control Models Technol. (SACMAT)*, Sophia Antipolis, France, Jun. 2007, pp. 133–138. [Online]. Available: http://doi.acm.org/10.1145/1266840.1266861

[17] E. Bertino, N. Shang, and S. S. Wagstaff, Jr., “An efficient time-bound hierarchical key management scheme for secure broadcasting,” *IEEE Trans. Dependable Secure Comput.*, vol. 5, no. 2, pp. 65–70, Apr./Jun. 2008. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/TDSC.2007.70241

[18] B. Toxen, “The NSA and Snowden: Securing the all-seeing eye,” *Commun. ACM*, vol. 57, no. 5, pp. 44–51, 2014.

[19] L. Qiu, Y. Zhang, F. Wang, M. Kyung, and H. R. Mahajan, “Trusted computer system evaluation criteria,” *Nat. Comput. Secur. Center*. vol. DoD 5200.28-STD, Dec. 1985. [Online]. Available: http://csrc.nist.gov/publications/history/dod85.pdf

[20] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. D. Keromytis, *The KeyNote Trust Management System, Version 2*, document IETF RFC 2704, Sep. 1999. [Online]. Available: https://tools.ietf.org/html/rfc2704

[21] D. E. Denning and M. Smid, “Key escrowing today,” *IEEE Commun. Mag.*, vol. 32, no. 9, pp. 58–68, Sep. 1994.

[22] A. Castiglione, L. Catuogno, A. D. Sorbo, U. Fiore, and F. Palmieri, “A secure file sharing service for distributed computing environments,” *J. Supercomput.*, vol. 67, no. 3, pp. 691–710, 2014. [Online]. Available: http://dx.doi.org/10.1007/s11227-013-0975-y

[23] A. De Santis, A. L. Ferrara, and B. Masucci, “Unconditionally secure key assignment schemes,” *Discrete Appl. Math.*, vol. 154, no. 2, pp. 234–252, 2006.

[24] G. Ateniese, A. De Santis, A. L. Ferrara, and B. Masucci, “A note on time-bound hierarchical key assignment schemes,” *Inf. Process. Lett.*, vol. 113, nos. 5–6, pp. 151–155, 2013.

[25] S. Goldwasser and S. Micali, “Probabilistic encryption,” *J. Comp. System Sci.*, vol. 28, no. 2, pp. 270–299, 1984.

[26] J. Katz and M. Yung, “Characterization of security notions for probabilistic private-key encryption,” *J. Cryptol.*, vol. 19, no. 1, pp. 67–95, 2006.

[27] O. Goldreich, S. Goldwasser, and S. Micali, “How to construct random functions,” *J. ACM*, vol. 33, no. 4, pp. 792–807, Aug. 1986.

[28] M. Blum and S. Micali, “How to generate cryptographically strong sequences of pseudorandom bits,” *SIAM J. Comput.*, vol. 13, no. 4, pp. 850–864, 1984.

[29] J. HÅstad, R. Impagliazzo, L. A. Levin, and M. Luby, “A pseudorandom generator from any one-way function,” *SIAM J. Comput.*, vol. 28, no. 4, pp. 1364–1396, 1999.

[30] M. Naor and O. Reingold, “Number-theoretic constructions of efficient pseudo-random functions,” *J. ACM*, vol. 51, no. 2, pp. 231–262, 2004.

[31] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway, “A concrete security treatment of symmetric encryption,” in *Proc. IEEE 38th Annu. Symp. Found. Comp. Sci.*, Oct. 1997, pp. 394–403.

[32] M. Bellare, R. Canetti, and H. Krawczyk, “Keying hash functions for message authentication,” in *Proc. 16th CRYPTO*, vol. 1109. 1996, pp. 1–15.

[33] A. Shamir, “How to share a secret,” *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[34] G. R. Blakley, “Safeguarding cryptographic keys,” in *Proc. AFIPS Nat. Comput. Conf.*, 1979, pp. 313–317.

[35] M. Ito, A. Saito, and T. Nishizeki, “Secret sharing scheme realizing general access structure,” in *Proc. IEEE Globecom*, Nov. 1987, pp. 99–102.

[36] J. Benaloh and J. Leichter, “Generalized secret sharing and monotone functions,” in *Proc. CRYPTO*, vol. 403. 1990, pp. 27–35.

[37] D. R. Stinson, “An explication of secret sharing schemes,” *Design, Codes Cryptogr.*, vol. 2, no. 4, pp. 357–390, 1992.

[38] R. M. Capocelli, A. De Santis, L. Gargano, and U. Vaccaro, “On the size of shares for secret sharing schemes,” *J. Cryptol.*, vol. 6, no. 3, pp. 157–167, 1993.

[39] C. Blundo, A. De Santis, and A. G. Gaggia, “Probability of shares in secret sharing schemes,” *Inf. Process. Lett.*, vol. 72, nos. 5–6, pp. 169–175, 1999.

[40] V. Daza, J. Herranz, P. Morillo, and C. Ràfols, “CCA2-secure threshold broadcast encryption with shorter ciphertexts,” in *Provable Security* (Lecture Notes in Computer Science), vol. 4784, W. Susilo, J. K. Liu, and Y. Mu, Eds. Wollongong, N.S.W., Australia: Springer, Nov. 2007, pp. 35–50. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-75670-5_3

[41] M. Bellare, A. Boldyreva, and S. Micali, “Public-key encryption in a multi-user setting: Security proofs and improvements,” in *Advances Cryptology*. Berlin, Germany: Springer, 2000, pp. 259–274.