Efficient Provably-Secure Hierarchical Key Assignment Schemes

Alfredo De Santis, Anna Lisa Ferrara, and Barbara Masucci

Dipartimento di Informatica ed Applicazioni, Università di Salerno
84084 Fisciano (SA), Italy

Abstract. A *hierarchical key assignment scheme* is a method to assign some private information and encryption keys to a set of classes in a partially ordered hierarchy, in such a way that the private information of a higher class can be used to derive the keys of all classes lower down in the hierarchy.

In this paper we design and analyze hierarchical key assignment schemes which are provably-secure and support dynamic updates to the hierarchy with local changes to the public information and without requiring any private information to be re-distributed.

- We first show an encryption based construction which is provably secure with respect to key indistinguishability, requires a single computational assumption and improves on previous proposals.
- Then, we show how to reduce key derivation time at the expense of an increment of the amount of public information, by improving a previous result.
- Finally, we show a construction using as a building block a public-key broadcast encryption scheme. In particular, one of our constructions provides constant private information and public information linear in the number of classes in the hierarchy.

1 Introduction

The *hierarchical access control problem* is defined in a scenario where the users of a computer system are organized in a hierarchy formed by a certain number of disjoint *security classes*. Hierarchical structures arise from the fact that some users have more access rights than others, and are widely employed in many different application areas, including database management systems, computer networks, operating systems, military, and government communications.

In 1983, Akl and Taylor [1] suggested the use of cryptographic techniques to enforce access control in hierarchical structures. In particular, they designed a *hierarchical key assignment scheme* where each class is assigned an encryption key that can be used, along with some public parameters, to compute the key assigned to all classes lower down in the hierarchy. This assignment is carried out by a Trusted Authority (TA), which is active only at the distribution phase. A recent work by Crampton et al. [10] provides a detailed classification of many schemes proposed in the last twenty years and evaluates their merits. Atallah

L. Kučera and A. Kučera (Eds.): MFCS 2007, LNCS 4708, pp. 371–382, 2007.

et al. [3,4] first addressed the problem of formalizing security requirements for hierarchical key assignment schemes. They proposed a first construction based on pseudorandom functions and a second one requiring the use of a symmetric encryption scheme secure against chosen-ciphertext attacks.

In this paper we design and analyze hierarchical key assignment schemes which are provably-secure and efficient. We consider security with respect to *key indistinguishability*, which corresponds to the requirement that an adversary is not able *to learn any information* about a key that it should not have access to. We propose two constructions for hierarchical key assignment schemes. Both constructions support updates to the access hierarchy with local changes to the public information and without requiring any private information to be re-distributed. The first construction, which is based on symmetric encryption schemes, is simpler than the one proposed by Atallah et al. [4], requires a single computational assumption, and offers more efficient procedures for key derivation and key updates. We also focus on improving efficiency of key derivation in hierarchical key assignment schemes. Such a problem has been recently considered by Atallah et al. [4,5], who proposed two different techniques requiring an increment of public information. We show how to further reduce key derivation time by improving one of their techniques. Finally, we show how to construct a hierarchical key assignment scheme by using only a public-key broadcast encryption scheme. In particular, by plugging in the scheme proposed by Boneh et al. [8] we obtain a hierarchical key assignment scheme offering constant private information and public information linear in the number of classes.

The full version of this paper, including a complete analysis and proofs, can be found in [11].

2 Hierarchical Key Assignment Schemes

Consider a set of users divided into a number of disjoint classes, called *security classes*. A binary relation $\preceq$ that partially orders the set of classes V is defined in accordance with authority, position, or power of each class in V. The poset $(V, \preceq)$ is called a *partially ordered hierarchy*. For any two classes u and v, the notation $u \preceq v$ is used to indicate that the users in v can access u's data. Clearly, since v can access its own data, it holds that $v \preceq v$, for any $v \in V$. We denote by A_v the set $\{u \in V : u \preceq v\}$, for any $v \in V$. The partially ordered hierarchy $(V, \preceq)$ can be represented by the directed graph $G^* = (V, E^*)$, where each class corresponds to a vertex in the graph and there is an edge from class v to class u if and only if $u \preceq v$. We denote by $G = (V, E)$ the *minimal representation* of the graph G^*, that is, the directed acyclic graph corresponding to the *transitive and reflexive reduction* of the graph $G^* = (V, E^*)$. Such a graph G has the same transitive and reflexive closure of G^*, i.e., there is a path (of length greater than or equal to zero) from v to u in G if and only if there is the edge (v, u) in E^*.

A hierarchical key assignment scheme for a family Γ of graphs, corresponding to partially ordered hierarchies, is defined as follows.

Definition 1. *A* hierarchical key assignment scheme *for Γ 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 and a graph $G = (V, E)$ in Γ, and produces as outputs*
   (a) *a private information s_u, for any class $u \in V$;*
   (b) *a key k_u, 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 and G, where s and k denote the sequences of private information and of keys, respectively.
2. *The* key derivation algorithm *Der is deterministic polynomial-time. It takes as inputs the security parameter 1^τ, a graph $G = (V, E)$ in Γ, two classes $u \in V$ and $v \in A_u$, the private information s_u assigned to class u and the public information pub, and produces as output the key k_v assigned to class v.*
   We require that for each class $u \in V$, each class $v \in A_u$, each private information s_u, each key k_v, each public information pub which can be computed by Gen on inputs 1^τ and G, it holds that $Der(1^\tau, G, u, v, s_u, pub) = k_v$.

A hierarchical key assignment scheme is evaluated according to several parameters, such as the amount of private information held by each user, the amount of public data, the complexity of key derivation, and the resistance to collusive attacks. In order to evaluate the security of the scheme, we consider a *static adversary* which wants to attack a class $u \in V$ and which is able to corrupt *all* users not allowed to compute the key k_u. We define an algorithm $Corrupt_u$ which, on input the private information s generated by the algorithm Gen, extracts the secret values s_v associated to all classes in the set $F_u = \{v \in V : u \notin A_v\}$. We denote by *corr* the sequence output by $Corrupt_u(s)$. 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 $1/2$.

If $A(\cdot, \cdot, \ldots)$ is any probabilistic algorithm then $a \leftarrow A(x, y, \ldots)$ denotes the experiment of running A on inputs $x, y, \ldots$ and letting a be the outcome, the probability being over the coins of A. Similarly, if X is a set then $x \leftarrow X$ denotes the experiment of selecting an element uniformly 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 $\epsilon : N \rightarrow R$ 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 2. `[IND-ST]` *Let Γ be a family of graphs corresponding to partially ordered hierarchies, let $G = (V, E)$ be a graph in Γ, let (Gen, Der) be a hierarchical key assignment scheme for Γ and let $\mathtt{STAT}_u$ be a static adversary which attacks a class u. Consider the following two experiments:*

Experiment $\mathbf{Exp}^{\mathtt{IND}-1}_{\mathtt{STAT}_u}(1^\tau, G)$	Experiment $\mathbf{Exp}^{\mathtt{IND}-0}_{\mathtt{STAT}_u}(1^\tau, G)$
$(s, k, pub) \leftarrow Gen(1^\tau, G)$	$(s, k, pub) \leftarrow Gen(1^\tau, G)$
$corr \leftarrow Corrupt_u(s)$	$corr \leftarrow Corrupt_u(s)$
$d \leftarrow \mathtt{STAT}_u(1^\tau, G, pub, corr, k_u)$	$\rho \leftarrow \{0,1\}^{length(k_u)}$
return d	$d \leftarrow \mathtt{STAT}_u(1^\tau, G, pub, corr, \rho)$
	return d

The advantage of $\mathtt{STAT}_u$ *is defined as*

$$\mathbf{Adv}^{\mathtt{IND}}_{\mathtt{STAT}_u}(1^\tau, G) = |Pr[\mathbf{Exp}^{\mathtt{IND}-1}_{\mathtt{STAT}_u}(1^\tau, G) = 1] - Pr[\mathbf{Exp}^{\mathtt{IND}-0}_{\mathtt{STAT}_u}(1^\tau, G) = 1]|.$$

The scheme is said to be secure in the sense of IND-ST *if, for each graph* $G = (V, E)$ *in* Γ *and each* $u \in V$*, the function* $\mathbf{Adv}^{\mathtt{IND}}_{\mathtt{STAT}_u}(1^\tau, G)$ *is negligible, for each static adversary* $\mathtt{STAT}_u$ *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. In [6] it has been proven that security against adaptive adversaries is (polynomially) equivalent to security against static adversaries. Hence, in this paper we will only consider static adversaries.

3 An Encryption Based Construction

In this section we consider the problem of constructing a hierarchical key assignment scheme by using as a building block a symmetric encryption scheme. A simple way to realize an encryption based scheme would be to assign a key k_u to each class $u \in V$ and a public information $p_{(u,v)}$, for each edge $(u, v) \in E$, corresponding to the encryption of k_v with the key k_u. This would allow any user in a class u to compute the key k_v held by any class v lower down in the hierarchy, by performing $dist_G(u, v)$ decryptions, where $dist_G(u, v)$ denotes the length of the shortest path from u to v in G.

Unfortunately, the simple solution described above is not secure with respect to *key indistinguishability*. Indeed, consider an adversary attacking a class u and corrupting a class v such that $(u, v) \in E$. The adversary, on input a challenge ρ, corresponding either to the key k_u or to a random value, is able to tell if ρ corresponds to the encryption key k_u simply by checking whether the decryption of the public value $p_{(u,v)}$ with key ρ corresponds to the key k_v held by class v. In order to avoid the above attack, we propose a new construction, described in Figure 1 and referred to as the *Encryption Based Construction (EBC)*, where the key assigned to a class is never used to encrypt the keys assigned to other classes. In particular, in the EBC each class $u \in V$ is assigned a private information s_u, an encryption key k_u, and a public information $\pi_{(u,u)}$, which is the encryption of the key k_u with the private information s_u; moreover, for each edge $(u, v) \in E$, there is a public value $p_{(u,v)}$, which allows class u to compute the private information s_v held by class v. Indeed, $p_{(u,v)}$ consists of the encryption of the private information s_v with the private information s_u. This allows any user in a class u to compute the key k_v held by any class v lower down in the hierarchy, by performing

Let Γ be a family of graphs corresponding to partially ordered hierarchies. Let $G = (V, E) \in \Gamma$ and let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme.

Algorithm $Gen(1^\tau, G)$

1. For any class $u \in V$, let $s_u \leftarrow \mathcal{K}(1^\tau)$ and $k_u \leftarrow \{0,1\}^\tau$;
2. Let s and k be the sequences of private information and keys, respectively, computed in the previous step;
3. For any two classes $u, v \in V$ such that $(u, v) \in E$, compute the public information $p_{(u,v)} = \mathcal{E}_{s_u}(s_v)$;
4. For any class u in V, compute the public information $\pi_{(u,u)} = \mathcal{E}_{s_u}(k_u)$;
5. Let *pub* be the sequence of public information computed in the previous two steps;
6. Output (s, k, pub).

Algorithm $Der(1^\tau, G, u, v, s_u, pub)$

1. Consider a path $(w_0, w_1), \ldots, (w_{m-1}, w_m) \in E$, from $u = w_0$ to $v = w_m$. For any $i = 1, \ldots, m$, extract the public value $p_{(w_{i-1}, w_i)}$ from *pub* and compute the private information $s_{w_i} = \mathcal{D}_{s_{w_{i-1}}}(p_{(w_{i-1}, w_i)})$;
2. Extract the public value $\pi_{(v,v)}$ from *pub* and output the key $k_v = \mathcal{D}_{s_v}(\pi_{(v,v)})$.

Fig. 1. The Encryption Based Construction (EBC)

$dist_G(u, v) + 1$ decryptions. We will show that an adversary attacking a class u 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. We first recall the definition of a symmetric encryption scheme.

Definition 3. *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* $\mathcal{D}(1^\tau, key, y) = m$*.*

Before analyzing the security of the EBC we first need to 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 [13]. 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 (`IND-P1-C0`), 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 $1/2$. The proof of the next theorem can be found in [11].

Theorem 1. *If the encryption scheme* $\Pi = (\mathcal{K}, \mathcal{D}, \mathcal{E})$ *is secure in the sense of* `IND-P1-C0`*, then the EBC is secure in the sense of* `IND-ST`*.*

The EBC requires $|E| + |V|$ public values; on the other hand, each class has to store a single secret value, corresponding to its private information. As for key derivation, a class $u \in V$ which wants to compute the key held by a class $v \in A_u$ is required to perform $dist_G(u, v) + 1$ decryption operations. The EBC supports insertions, but not deletions, of classes and edges in the hierarchy without redistributing private information to the classes affected by such changes. However, in the full version of this paper [11] a simple modification of the scheme, which avoids such a re-distribution, has been proposed. The modified scheme, referred to as the *Dynamic Encryption Based Construction (DEBC)*, requires $|E| + 2|V|$ public values, whereas, each class has to store a single secret value. Moreover, the number of decryptions needed by class u to compute the key of class $v \in A_u$ is $dist_G(u, v) + 2$.

4 Improving Key Derivation Time

In the EBC, as well as in the schemes proposed by Atallah et al. [3,4], the number of steps that a class u has to perform, in order to compute the key of another class v lower down in the hierarchy, is proportional to the length of the shortest path from u to v. Atallah et al. [3,4,5] analyzed the problem of reducing key derivation time by modifying the graph representing the hierarchy, in order to reduce its diameter, where the *diameter* of a directed graph is defined as the maximum distance between a pair of vertices in the graph connected by a path. To this aim, they proposed some constructions to add additional edges, called *shortcut edges*, as well as *dummy vertices*, to the hierarchy.

4.1 The Shortcutting Technique

The *shortcutting* of a directed graph $G = (V, E)$ consists into inserting *shortcut edges* in E, without changing the transitive closure of G. The goal is to obtain another directed graph, called a *shortcut graph*, having a smaller diameter than G.

The shortcutting technique is quite old, indeed it has been first considered in 1982 by Yao [17]. In particular, Yao considered the problem in a quite different

context, where the n elements of V belong to a given semigroup $(S, \circ)$ and one is interested in answering queries of the form *"what is the value of* $v_i \circ v_{i+1} \circ \cdots \circ v_{j-1} \circ v_j$*?"* for any $1 \leq i \leq j \leq n$. In the following we translate to our scenario the main existing results concerning the shortcutting technique when applied to particular kinds of graphs. We start discussing chains, then we analyze trees and finally general graphs.

Chains. By using the techniques proposed by Yao [17] we can add shortcut edges to a chain $(v_1, \ldots, v_n)$ of n vertices. The techniques proposed by Alon and Schieber [2] in 1987 and Bodlaender et al. [7] in 1994 are essentially the same as the ones proposed by Yao, but their description is easier to illustrate. The details of the constructions, translated to our scenario, can be found in [11]. The parameters of such constructions are summarized in Figure 2, where $\log^* n$, is the *iterated logarithmic function.*

Trees. In 1987 Chazelle [9], as well as Alon and Schieber [2], considered the problem of reducing the diameter of free trees, i.e., indirected connected acyclic graphs, by adding shortcut edges. Their results, which are summarized in Figure 2, were also shown to hold for directed trees [16].

Diameter ℓ	Minimal number of shortcut edges
1	$\Theta(n^2)$
2	$\Theta(n \cdot \log n)$
3	$\Theta(n \cdot \log \log n)$
4	$\Theta(n \cdot \log^* n)$
$O(\log^* n)$	$\Theta(n)$

Fig. 2. Minimal number of shortcut edges to be added to chains and trees with n vertices in order to obtain a shortcut graph with diameter ℓ

General Graphs. Thorup [15] conjectured that for any directed graph $G = (V, E)$ one can obtain a shortcut graph with diameter polylogarithmic in $|V|$, i.e., $(\log |V|)^{O(1)}$, by adding at most $|E|$ shortcut edges. He also showed his conjecture to be true for planar directed graphs [16]. However, Hesse [14] gave a counterexample to Thorup's conjecture. He showed how to construct a direct graph requiring the addition of $\Omega(|E| \cdot |V|^{1/17})$ shortcut edges to reduce its diameter below $\Theta(|V|^{1/17})$. By extending his construction to higher dimensions, it is possible to obtain graphs with $|V|^{1+\epsilon}$ edges that require the addition of $\Omega(|V|^{2-\epsilon})$ shortcut edges to reduce their diameter.

All constructions described in this section can be used to reduce key derivation time in hierarchical key assignment schemes. However, the result by Hesse [14] implies that key derivation time cannot be reduced *essentially* below $\Omega(|V|^2)$ for some kinds of graphs by adding only shortcut edges.

4.2 The Shortcutting and Point-Inserting Technique

Atallah et al. [5] also proposed a different technique to reduce the diameter of an access hierarchy. Such a technique consists of the addition of *dummy vertices*, as well as new edges, to the hierarchy. The idea is to obtain a new hierarchy such that there exists a path between two classes u and v in the old hierarchy if and only if there exists a path between u and v in the new one. The addition of dummy vertices results in a smaller number of new edges to be added to the hierarchy.

The technique makes use of the concept of *dimension* of a poset, originally defined by Dushnik and Miller [12]. The dimension of a poset $(V, \preceq)$ is the minimum number of total orders on V whose intersection is $(V, \preceq)$. It can also be seen as the smallest nonnegative integer d for which each $u \in V$ can be represented by a d-vector $(x_{u,1}, \ldots, x_{u,d})$ of integers such that $u \preceq v$ if and only if $x_{u,i} \leq x_{v,i}$, for any $i = 1, \ldots, d$, and any $u, v \in V$. There are efficient algorithms to test if a poset has dimension 1 or 2, but the problem of determining if a poset has dimension 3 is NP-complete. A poset has dimension one if and only if it is a total order.

When applied to a hierarchy with n classes and dimension d, the technique allows to reduce key derivation time to $2d + 1$, by adding $O(n \cdot d \cdot (\log n)^{d-1} \cdot \log \log n)$ dummy classes and new edges. In the following we show how to further reduce key derivation time. Our technique performs a further shortcutting of the graph obtained by Atallah et al.'s technique and allows key derivation time to be independent on d.

4.3 The Improved Shortcutting and Point-Inserting Technique

In this section we consider the problem of reducing the diameter of the graph obtained by the shortcutting and point-inserting technique, on input a poset $(V, \preceq)$ with dimension d. Our construction, which we refer in the following as the *Improved Shortcutting and Point-Inserting Technique (ISPIT)* is recursive, and for the base case $d = 1$ reduces to the construction proposed by Yao [17]. The construction for the case $d \geq 2$ is described in Figure 3. The input is a set of n d-dimensional points corresponding to the vertices in V; for each vertex $v \in V$, let $P_v^{(d)}$ be the corresponding point and let $V^{(d)} = \{P_v^{(d)} : v \in V\}$.

The number $DP(n, d)$ of dummy points added by the ISPIT is $DP(n, d) = 2 \cdot DP(\lceil n/2 \rceil, d) + DP(n, d-1) + \Theta(n)$, where $DP(n, 1) = 0$ and $DP(1, d) = 0$. Indeed, in order to construct $G^{(d)}$, the algorithm adds n dummy points, corresponding to the projections of the points in $V^{(d)}$ on the $(d-1)$-dimensional hyperplane M, plus $DP(n, d-1)$ dummy points for the construction of $G^{(d-1)}$, and then is recursively called on the two sets $V_1^{(d)}$ and $V_2^{(d)}$. The solution of the above recurrence is $DP(n) = \Theta(n \cdot d \cdot (\log n)^{d-1})$. On the other hand, the number $T(n)$ of new edges added by the ISPIT is $T(n, d) \leq 2 \cdot T(\lceil n/2 \rceil, d) + 3 \cdot T(n, d-1) + \Theta(n)$, where $T(n, 1)$ denotes the number of shortcut edges added by Yao's construction [17] for the case $d = 1$ in order to obtain a shortcut graph having a certain diameter, whereas, $T(1, d) = 0$. Indeed, at most $3 \cdot |E^{(d-1)}| + n$ new edges are added in steps 7. and 8. and then the algorithm is recursively called on the two sets $V_1^{(d)}$ and $V_2^{(d)}$. Clearly, the solution of $T(n, d)$, as well as the diameter of the graph

Let $(V, \preceq)$ be a poset with dimension $d \geq 2$, let $V^{(d)}$ be the set of points in the vectorial representation of the Hasse diagram associated to $(V, \preceq)$ and based on its d total orders, and let $\ell \geq 1$.

1. If $|V^{(d)}| = 1$, then output $V^{(d)}$.
2. If $|V^{(d)}| \geq 2$, compute a $(d-1)$-dimensional hyperplane M perpendicular to the d-th dimension that partitions the set of points in $V^{(d)}$ into two sets $V_1^{(d)}$ and $V_2^{(d)}$ of $\lfloor n/2 \rfloor$ and $\lceil n/2 \rceil$ points, respectively, where $V_1^{(d)}$ is the set on the smaller side of the hyperplane (according to the d-th coordinate). Such points are projected on M. Denote by $P_v^{(d-1)}$ the projection of $P_v^{(d)}$ on M. Let $V_1^{(d-1)}$ and $V_2^{(d-1)}$ be the projections of $V_1^{(d)}$ and $V_2^{(d)}$.
3. If $d = 2$, use Yao's construction on the chain whose vertices are the points in the set $V^{(1)}$, in order to obtain a shortcut graph $G^{(1)} = (V^{(1)}, E^{(1)})$, having diameter at most ℓ. The set of dummy points added by the algorithm is $D^{(1)} = \emptyset$ (no dummy points are added).
4. If $d \geq 3$, recursively call the algorithm on the set of points in $V^{(d-1)} = V_1^{(d-1)} \cup V_2^{(d-1)}$, corresponding to a $(d-1)$-dimensional hyperplane; let $G^{(d-1)} = (V^{(d-1)} \cup D^{(d-1)}, E^{(d-1)})$ be the corresponding output.
5. Let $D^{(d)} = V^{(d-1)} \cup D^{(d-1)}$.
6. Let $E^{(d)} = E^{(d-1)}$.
7. Add edges between points in $V^{(d)}$ and corresponding projections:
   (a) For each point $P_v^{(d)} \in V_1^{(d)}$, add an edge $(P_v^{(d-1)}, P_v^{(d)})$ to $E^{(d)}$.
   (b) For each point $P_v^{(d)} \in V_2^{(d)}$, add an edge $(P_v^{(d)}, P_v^{(d-1)})$ to $E^{(d)}$.
8. Add shortcut edges between points in $V^{(d)}$ and dummy points:
   (a) For each edge $(P_u^{(d-1)}, P_v^{(j)}) \in E^{(d-1)}$, add an edge $(P_u^{(d)}, P_v^{(j)})$ to $E^{(d)}$.
   (b) For each edge $(P_u^{(j)}, P_v^{(d-1)}) \in E^{(d-1)}$, add an edge $(P_u^{(j)}, P_v^{(d)})$ to $E^{(d)}$.
9. Recursively call the algorithm on the two sets of points in $V_1^{(d)}$ and $V_2^{(d)}$.
10. Output the graph $G^{(d)} = (V^{(d)} \cup D^{(d)}, E^{(d)})$.

Fig. 3. The Improved Shortcutting and Point-Inserting Technique (ISPIT)

$G^{(d)}$, depends on the the number $T(n, 1)$ of shortcut edges added by Yao's construction. If $T(n, 1) = \Theta(n)$, then $T(n, d) = O(n \cdot d \cdot (3 \log n)^{d-1})$. On the other hand, if $T(n, 1) = \Theta(n \cdot \log \log n)$, then $T(n, d) = O(n \cdot d \cdot (3 \log n)^{d-1} \cdot \log \log n)$ and the diameter of the graph $G^{(d)}$ is three, i.e., it is independent on d. It is easy to see that, for any two vertices $u, v \in V$ such that $u \preceq v$, there exists a path from $P_v^{(d)}$ to $P_u^{(d)}$ in $G^{(d)}$ which has length at most the diameter of the graph $G^{(1)}$ obtained by solving the 1-dimensional problem on $V^{(1)}$.

Compared to the technique in [5], the ISPIT allows a further reduction of the diameter, but in each recursive call, it adds at most three times the number of new edges added by that algorithm. In the following we show a trade-off between the number of edges added by the ISPIT and the diameter of the resulting graph. The idea behind the construction is the following: Assume the 1-dimensional problem is solved by adding $\Theta(n \log \log n)$ shortcut edges. For each $j = 2, \ldots, d$, the j-dimensional problem could be solved either with the technique in [5] or with

ours. Let $1 \leq t \leq d$ and assume, for example, that for $j = 2, \ldots, t$, the technique in [5] is used to solve the j-dimensional problem, whereas, our technique is used to solve the problems with dimensions from $t+1$ to d. It is easy to see that the graph resulting by the above construction has diameter $2t+1$. Moreover, the number of new edges added by the modified algorithm is $O(3^{d-t} \cdot n \cdot t \cdot (\log n)^{d-1} \cdot \log \log n)$.

5 A Broadcast Encryption Based Construction

In this section we show how to construct a hierarchical key assignment scheme using as a building block a broadcast encryption scheme. 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 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. In contrast, in the public key setting a public key published by a trusted authority allows anybody to broadcast a message. We first recall the definition of a public-key broadcast encryption scheme [8].

Definition 4. *A* public-key broadcast encryption scheme *for a set $\mathcal{U}$ of users is a triple of algorithms (Set, Enc, Dec) satisfying the following conditions:*

1. *The* setup algorithm *Set is probabilistic polynomial-time. It takes as input a security parameter 1^τ and the set of users $\mathcal{U}$ and produces as output a private key sk_u, for each user $u \in \mathcal{U}$, and a public key pk.*
2. *The* encryption algorithm *Enc is probabilistic polynomial-time. It takes as inputs 1^τ, a subset $X \subseteq \mathcal{U}$, and the public key pk, and produces as output a pair (Hdr, k), where Hdr is called the broadcast header and k is a encryption key. Let m be a message to be broadcast in such a way that only users in X are allowed to obtain m and let y be the encryption of m under the symmetric key k. The broadcast message consists of (X, Hdr, y), where the pair (X, Hdr) is called the full broadcast header and y is called the broadcast body.*
3. *The* decryption algorithm *Dec is deterministic polynomial-time. It takes as inputs 1^τ, a subset $X \subseteq \mathcal{U}$, a user $u \in X$ and its private key sk_u, a broadcast header Hdr, and the public key pk, and produces as output the key k. Such a key can be used to decrypt the broadcast body y in order to obtain m.*

We require that for all subsets $X \subseteq \mathcal{U}$, all users $u \in X$, all public keys and private keys which can be output by $Set(1^\tau, \mathcal{U})$, all pairs (Hdr, k), which can be output by $Enc(1^\tau, X, pk)$, we have that $Dec(1^\tau, X, u, sk_u, Hdr, pk) = k$.

The idea behind our construction, referred in the following as the *Broadcast Encryption Based Construction (BEBC)*, is to compute the private and public information by using the broadcast encryption scheme; more precisely, the public information will contain a broadcast header Hdr_u, which corresponds to an encryption of the key k_u, for each class $u \in V$. Such a broadcast header can be decrypted by all classes in the set $I_u = \{v \in V :$ there is a path from v to u in $G\}$, allowing them to compute the key k_u. The Broadcast Encryption Based Construction is described in Figure 4.

Let Γ be a family of graphs corresponding to partially ordered hierarchies. Let $G = (V, E) \in \Gamma$ and let (Set, Enc, Dec) be a public-key broadcast encryption scheme for users in V.

Algorithm $Gen(1^\tau, G,)$

1. Run $Set(1^\tau, V)$ to generate a public key pk and a secret key sk_u for any $u \in V$;
2. For each class $u \in V$, let $s_u = sk_u$;
3. For each class $u \in V$, run $Enc(1^\tau, I_u, pk)$ to obtain the pair (Hdr_u, k_u);
4. Let s and k be the sequences of private information and keys computed in the previous two steps;
5. Let pub be the sequence constituted by the public key pk along with the header Hdr_u, for any $u \in V$;
6. Output (s, k, pub).

Algorithm $Der(1^\tau, G, u, v, s_u, pub)$

1. Extract the public key pk and the header Hdr_v from pub.
2. Output $k_v = Dec(1^\tau, I_v, u, s_u, Hdr_v, pk)$.

Fig. 4. The Broadcast Encryption Based Construction

Before analyzing the security of the BEBC we first need to define what we mean by a *secure* public-key broadcast encryption scheme. The security of a public-key broadcast encryption scheme is defined through a game between an adversary A and a challenger. According to the capabilities of the adversary and the security goal, several types of security notions for public-key broadcast can be defined. We consider the definition of *semantic security* given by Boneh et al. [8], where the adversary is not allowed to issue decryption queries to the challenger. We consider the following game: First, algorithm A outputs a set $X \subseteq \mathcal{U}$ of receivers that it wants to attack. Then, the challenger first runs $Set(1^\tau, \mathcal{U})$ to obtain a private key sk_u for each user $u \in \mathcal{U}$ and a public key pk. Afterwards, it gives the public key pk and all private keys sk_v for which $v \notin X$ to A. The challenger runs $Enc(1^\tau, X, pk)$ to obtain (Hdr, k). Then, it picks a random bit $b \in \{0, 1\}$, sets $k_b = k$ and chooses $k_{\overline{b}}$ as a random key. The challenge (Hdr, k_0, k_1) is given to A. Algorithm A outputs its guess $b' \in \{0, 1\}$ for b and wins the game if $b = b'$. The advantage of the adversary A is defined as $\mathbf{Adv}_{A,\mathcal{U}}(1^\tau) = |Pr[b' = b] - 1/2|$. The scheme is said to be *semantically secure* if the function $\mathbf{Adv}_{A,\mathcal{U}}(1^\tau)$ is negligible, for any adversary A whose time complexity is polynomial in τ. The proof of the next theorem can be found in [11].

Theorem 2. *If the public-key broadcast encryption scheme* (Set, Enc, Dec) *is semantically secure, then the BEBC is secure in the sense of* IND-ST.

Boneh et al. [8] showed how to construct a semantically secure public-key broadcast encryption scheme for a set of n users, assuming the intractability of the *n-Bilinear Decisional Diffie-Hellman Exponent* (n-BDDHE). The use of such a

broadcast encryption scheme allows us to obtain a hierarchical key assignment scheme where the public information consists of $4|V|+1$ group elements, whereas, the private information has constant size. Moreover, key derivation requires a single (complex) decryption operation, which involves at most $|V|-2$ group operations. The above scheme supports dynamic changes to the hierarchy without requiring re-distribution of private information to the classes affected by such changes. Details of the construction can be found in [11].

References

1. Akl, S.G., Taylor, P.D.: Cryptographic Solution to a Problem of Access Control in a Hierarchy. ACM Trans. on Comput. Syst. 1(3), 239–248 (1983)
2. Alon, N., Schieber, B.: Optimal Preprocessing for Answering On-line Product Queries, Tech. Rep, TR 71/87, Inst. of Comput. Sci., Tel-Aviv Univ. (1987)
3. Atallah, M.J., Frikken, K.B., Blanton, M.: Dynamic and Efficient Key Management for Access Hierarchies. In: Proc. of ACM CCS 2005, pp. 190–201(2005)
4. Atallah, M.J., Blanton, M., Fazio, N., Frikken, K.B.: Dynamic and Efficient Key Management for Access Hierarchies, CERIAS Tech. Rep. TR 2006-09, Purdue Univ. (2006)
5. Atallah, M.J., Blanton, M., Frikken, K.B.: Key Management for Non-Tree Access Hierarchies. In: Proc. of ACM SACMAT 2006, pp. 11–18, Full version avail at `http://www.cs.purdue.edu/homes/mbykova/papers/key-derivation.pdf`
6. Ateniese, G., De Santis, A., Ferrara, A.L., Masucci, B.: Provably-Secure Time-Bound Hierarchical Key Assignment Schemes. In: Proc. of ACM CCS 2006, pp. 288–297. Full version avail. as Rep. 2006/225 at the IACR Cryptology ePrint Archive (2006)
7. Bodlaender, H.L., Tel, G., Santoro, N.: Trade-offs in Non-reversing Diameter. Nordic J. on Comput. 1, 111–134 (1994)
8. Boneh, D., Gentry, C., Waters, B.: Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 258–275. Springer, Heidelberg (2005)
9. Chazelle, B.: Computing on a Free Tree via Complexity-Preserving Mappings. Algorithmica 2, 337–361 (1987)
10. Crampton, J., Martin, K., Wild, P.: On Key Assignment for Hierarchical Access Control. In: Proc. of IEEE CSFW, pp. 98–111 (2006)
11. De Santis, A., Ferrara, A.L., Masucci, B.: Efficient Provably-Secure Hierarchical Key Assignment Schemes, avail. as Rep. 2006/479 at the IACR Cryptology ePrint Archive (2006)
12. Dushnik, B., Miller, E.W.: Partially Ordered Sets. American Journal of Mathematics 63, 600–610 (1941)
13. Goldwasser, S., Micali, S.: Probabilistic Encryption. Journal of Comp. and System Sci. 28, 270–299 (1984)
14. Hesse, W.: Directed Graphs Requiring Large Number of Shortcuts. In: Proc. of ACM-SIAM SODA 2003, pp. 665–669 (2003)
15. Thorup, M.: On Shortcutting Digraphs. In: Mayr, E.W. (ed.) WG 1992. LNCS, vol. 657, pp. 205–211. Springer, Heidelberg (1993)
16. Thorup, M.: Shortcutting Planar Digraphs. Combinatorics, Probability & Comput. 4, 287–315 (1995)
17. Yao, A.C.: Space-Time Tradeoff for Answering Range Queries. In: Proc. of ACM STOC 1982, pp. 128–136 (1982)