Efficient Multi-Dimensional Key Management in Broadcast Services*

Marina Blanton[1] and Keith B. Frikken[2]

[1] Department of Computer Science and Engineering, University of Notre Dame
mblanton@nd.edu
[2] Computer Science and Software Engineering, Miami University
frikkekb@muohio.edu

Abstract. The prevalent nature of Internet makes it a well suitable medium for many new types of services such as location-based services and streaming content. Subscribers to such services normally receive encrypted content and can obtain access to it if they possess the corresponding decryption key. Furthermore, in location-based services a subscription is normally granted to a geographic area specified by user-specific coordinates (x_1, x_2), (y_1, y_2) and custom time interval (t_1, t_2). Similarly, subscriptions to other services also involve multiple dimensions. The problem of key management is then to assign keys to each point on a D-dimensional grid and to subscribers in such a way as to permit all users to obtain access only to the resources in their subscriptions and minimize the associated overhead. In this work, we develop a novel key management scheme for multi-dimensional subscriptions that both outperforms existing solutions and supports a richer set of access privileges than existing schemes. Our scheme is provably secure under the Decision Linear Diffie-Hellman Assumption.

1 Introduction

The ubiquity of digital communication today allows it to be easily used for a variety of broadcast or streaming services. For instance, location-based services (LBS) have become widely spread and deployed; examples of such services include LOC-AID [2] and Garmin [1]. In these systems, a user typically can subscribe to a geo-spatial area for a specified duration of time and is able to query the system for spatial-temporal information such as traffic conditions, points of interest near a particular geographic location, or receive periodic updates such as the weather forecast. There is obviously a need to ensure that only legitimate subscribers can obtain access to the information within their subscription rights. Similarly, geographic information systems (GIS) collect and store large amounts of geo-spatial data such as satellite images, and there is a need to protect this data from unauthorized access. In particular, the importance of fine-grained access control mechanisms that would permit precise release of geo-spatial information was discussed in the NRC's IT roadmap to geo-spatial future [17] as a major challenge.

In such systems, a user typically subscribes for a fee to an area bounded by coordinates (x_1, y_1) and (x_2, y_2) for a specific time interval $[t_1, t_2]$. A user is then allowed

* Portions of this work were sponsored by AFOSR grant AFOSR-FA9550-09-1-0223 and NSF grant CNS-0915843.

to access the resource or broadcast associated with the coordinate (x, y) at time t if and only if $x_1 \leq x \leq x_2$, $y_1 \leq y \leq y_2$, and $t_1 \leq t \leq t_2$. The space is modeled as a two-dimensional grid of size $T_1 \times T_2$ cells. Time is also partitioned into small slots and becomes the third dimension of the grid. More generally, a grid of any dimension can be specified and used.

A typical solution in broadcast services is to encrypt content and broadcast the encrypted content. The access control is then enforced by distributing certain secret keys to subscribers and ensuring that the decryption key for a broadcasted resource is available only to the parties who are authorized to access the resource. This setup has several advantages including the ability to outsource the storage and distribution of encrypted content to a third-party provider, which is important in GIS systems or location-based services that deal with large volumes of data. Another advantage is that users can also remain anonymous while accessing an (authorized) resource.

Access control enforcement is therefore implemented via key management, which is a well studied topic. With this enforcement mechanism, the service provider assigns keys to the resources (in our context, all resources associated with a single position in a multi-dimensional space are assigned the same key). When a user subscribes to a set of resources in the system (i.e., a sub-grid in multi-dimensional space), she obtains secret information that will allow her to obtain access to the subscribed resources. This secret information either can directly contain decryption keys for all resources to which the user is entitled to have access or can permit derivation of all such keys. The challenge in designing such solutions is in ensuring that the access control policy is enforced while achieving the best possible performance. Therefore, the overhead of such solutions is measured in terms of the size of user secret keys, work necessary to derive a key, amount of additional information the service provider must maintain, etc.

While key management for dynamic groups or hierarchical systems (such as RBAC) is well-studied (see Section 2), solutions for geo-spatial systems and higher dimensions appeared only in the recent years [5, 20, 24]. In general, solutions that can be applied to a space of an arbitrary dimension D are desirable as they will allow any service or system to be used within this framework. As an example of an application that benefits from a key management scheme that supports any number of dimensions, consider streaming television. When a user subscribes to television, the access policy could be specified over a potentially large number of dimensions such as (i) content being accessed (at the level of station, show, or episode), (ii) quality of channel (e.g, data quality and commercial content), (iii) time of access (e.g., hour, day, month, and year), (iv) location of user (e.g., state and zipcode). Unlike subscribing to a rectangular region in location-based services, this subscription may not consist of a range over the dimensions; that is, a user could subscribe to multiple stations. Furthermore, more flexible subscriptions than continuous range in all dimensions can be desirable for location-based services as well: for example, there might be need to exclude certain geographic regions such as military installations or critical infrastructures from subscriptions.

Our contributions. We propose a novel key management scheme for multi-dimensional grids with attractive performance characteristics. In particular, it has the following properties, which none of the existing schemes can simultaneously achieve:

– Users can subscribe to an arbitrary set of points or intervals in each dimension, i.e., the subscription region does not have to be contiguous.
– The amount of user secret storage and the amount of work a user must perform to derive a key do not exponentially depend on the number of dimensions D; both storage and work are linear in D. This means that the scheme can be efficiently realized for applications where the number of dimensions is high without burdening the user; this substantially improves the performance compared to other schemes.
– Users do not need to access any external (publicly available) data for the purposes of aiding key derivation; broadcast content is all a user receives.
– The service provider needs to store only a constant amount of information associated with the scheme.

A more detailed comparison with prior literature is provided in the next section. We are able to achieve this performance by issuing sub-keys to users for each dimension separately and using a mechanism for tying the sub-keys of each user together to be able to maintain security (i.e., to achieve resilience against collusion). Our scheme enjoys provable security under the standard Decision Linear assumption.

2 Related Work

Related work on key management can be divided into two lines of research that go under the names of key management for access hierarchies and group key management. We give a brief overview of each of them next.

In hierarchical access control schemes, all users are divided into a set of access classes, which are organized in a hierarchy. Resources associated with each access class are encrypted with the corresponding encryption key. A user with access to a specific class is allowed to access resources at her own class and all descendant classes in the hierarchy. In order to lower overhead of such schemes, public information that helps in the key derivation process is used. Users from different classes use different parts of the public information data structure to derive necessary keys efficiently. Performance of such schemes is measured in terms of the number of keys a user stores, the size of public information, work needed to derive a key, and overhead associated with user joins and leaves.

The formal definitions of security in this context were put forward by Atallah et al. [7, 3] (the overall literature is very extensive, see, e.g., [7] for an overview), and, in particular, that work defined the notion of *key recovery* and *key indistinguishability* for key management schemes. Consequently, the work of Ateniese et al. [8] extended the definitions to time-based key management for a hierarchy of access classes, where time is partitioned into small slots and a user obtains access to a certain class in the hierarchy (and consequently to all descendant classes) for a certain contiguous interval of time which may differ for each user.[3] The authors also showed that the security notions in the presence of *static* and *adaptive* adversaries are (polynomial time) equivalent for time-based schemes, which means that showing security against static adversaries is sufficient for such schemes. In this extended framework, key derivation is now performed

[3] This problem was studied prior to Ateniese et al. [8] (see, e.g., [21, 15, 22]), but earlier schemes lack formal proofs of security and some of them are known to have security flaws.

for the purposes of hierarchical access control and time-based (i.e., one-dimensional) access control. Other work on time-based key management for user hierarchies includes [6, 12] that improve performance of the initial solutions in [4], lowering the overhead associated with the schemes. These latter publications give a mechanism for performing time-based key management that can be combined with any suitable hierarchical key management scheme, i.e., the mechanisms for achieving two goals are decoupled. More recently, techniques for higher dimensions were proposed as well. In particular, [5] gives an efficient solution for geo-spatial (i.e., two-dimensional) access control, which is further improved and extended to a higher number of dimensions in [24].

Literature on group key management is also concerned with the problem of key assignment to users and resources. No relationship between user classes or groups is assumed (i.e., key management is performed for each group independently), and instead the need to perform key derivation comes only from the dynamic nature of groups, where a user can join and leave a group at any time. The key difference between this line of work and work on hierarchical key management is (i) absence of relationship between the groups, and (ii) inability to use public storage. In particular, the use of public information greatly aids the performance of hierarchical schemes (including extensions to multiple dimensions) resulting in low overheads in terms of the number of user keys and key derivation time. In group key management protocols, it is assumed that some content is broadcast to the users, and the users can derive the necessary decryption key (using the broadcast and stored keys) if and only if they are authorized to access the content.

This problem is well studied with many solutions available (see, e.g., [14, 13, 23, 18]). Srivatsa et al. [20] were first to extend the framework to multiple dimensions, to enable such schemes to be used with location-based services such as spatial-temporal authorizations and subscription services of any dimensionality in general. This work provides a solution which is significantly more efficient than the straightforward use of prior group key management protocols for a single group, and supports access to a contiguous interval in each dimension. Its user overhead (i.e., the number of keys and key derivation time), however, is exponential in the number of dimensions, which makes it less suitable for applications where the number of dimensions is large. Our solution simultaneously removes exponential dependence on the number of dimensions (all overhead is at most linear in the number of dimensions) and improves expressiveness of the scheme by permitting user access to any subset of slots in each dimension.

We summarize performance of other solutions and our scheme in Table 1. In the table, D denotes the number of dimensions, X_i denotes the set of user subscription units in dimension i, where $|X_i|$ is the size of the set, and T_i denotes the maximum number of units in dimension i. The expressiveness column indicates whether a scheme supports only contiguous intervals in each dimension (in which case X_i can be specified as a range $[a, b]$ for $a \leq b$) or any subset of units in each dimension from the available range $[1, T_i]$. The communication overhead column indicates the amount of data that must be made available to permit key derivation for *all* authorized users when encrypted content for to a single point in the D-dimensional space is broadcast. That is, the solution of Yuan-Atallah uses a public data structure of the specified size that allows any user to efficiently derive decryption keys for any subscribed point in the D-dimensional space,

Scheme	User's keys	Key derivation	Comm. overhead	Expressiveness
Yuan-Atallah [24]	$O(1)$	$O(1)$	$O(\prod_{i=1}^{D} T_i \cdot (\log^* \log^* (\prod_{i=1}^{D} T_i))^D)$	contiguous interval
Srivatsa et al. [20]	$O(\frac{1}{D}(2^{D-1} \cdot \sum_{i=1}^{D} \log \lvert X_i \rvert))$	$O(\frac{1}{D}(2^{D} \cdot \sum_{i=1}^{D} \log \lvert X_i \rvert))$	–	contiguous interval
Our scheme	$O(\sum_{i=1}^{D} \lvert X_i \rvert)$	$O(D)$	$O(D)$	any subset

Table 1. Comparison with prior work.

Scheme	Public-key size	Enc. cost	Ciphertext size	Dec. key size	Dec. cost
Boneh-Waters [11]	$O(D \cdot T)$	$O(D \cdot T)$	$O(D \cdot T)$	$O(D)$	$O(D)$
Shi et al. [19]	$O(D \log T)$	$O(D \log T)$	$O(D \log T)$	$O(D \log T)$	$O((\log T)^D)$

Table 2. Performance of multi-dimensional query over encrypted data schemes.

but the solution is not well suited for uni-directional broadcast services since different users will need to use different parts of the public data structure. In our case, each encrypted transmission can be easily prepended with D data items which will permit all authorized users to obtain access to the content.

Another direction of research related to this work is queries over encrypted data. In particular, we mention the work of Shi et al. [19] on multi-dimensional range queries and the work of Boneh and Waters [11] that permits multi-dimensional subset and range queries. Since these schemes do not use key derivation (and therefore have different characteristics), but could potentially be used in our context, we provide their performance separately in Table 2. This table uses T as the number of points in each dimension, i.e., $T = T_1 = \ldots = T_D$. It is clear that in our context transmitting ciphertext of size $O(D \cdot T)$ (as in [11]) or having decryption cost of $O((\log T)^D)$ operations (as in [19]) is not acceptable. The higher computational cost in these schemes is dictated by stronger privacy properties (i.e., the inability to determine attributes associated with a ciphertext), which is not needed in our context.

Finally, we mention attribute-based encryption (ABE) as a potential realization of the functionality we seek. With traditional ABE, we will be able to form a ciphertext with D attributes which corresponds to a cell in the multi-dimensional grid. A client who wishes to subscribe to items $X = X_1 \times \cdots \times X_D$ will then have to store $\prod_{i=1}^{D} |X_i|$ keys (i.e, a key per cell of its subscription). Communication cost is $O(D)$ and decryption cost is also $O(D)$. If we employ a hierarchical ABE, the efficiency can potentially be improved through derivation, but the costs are still significant. If, for example, we use Boneh et al. [9] hierarchical identity based encryption (HIBE), which has performance characteristics among the best known for HIBE schemes in that the ciphertext size is independent of the number of elements T_i in each dimension, permitting a user to subscribe to only continuous intervals X_i in each dimension already leads to $O(\prod_{i=1}^{D}(\log |T_i| \log |X_i|))$ private key storage, and supporting arbitrary X_i's results in $O(\prod_{i=1}^{D}(|X_i| \log T_i))$ key material.[4] In such schemes, the size of each decryption

[4] This can be somewhat decreased with longer ciphertexts (e.g., of size $O(\prod_{i=1}^{D} \log T_i)$).

key depends on the height of the hierarchy, and $O(\prod_{i=1}^{D} \log |X_i|)$ ($O(\prod_{i=1}^{D} |X_i|)$) keys are needed to represent X in the case of contiguous intervals (resp., any subsets).

3 Model Description and Definitions

System Model. A service provider has a resource, which is associated with a point in D-dimensional space. We denote the number of items/intervals in jth dimension, for $j = 1, \ldots, D$, by T_j. We will assume that the units are numbered 1 through T_j, i.e., lie in the interval $[1, T_j]$. Then access to a resource with coordinates $(i_1, i_2, \ldots, i_D)$ in D-dimensional space will be secured using a cryptographic key $k_{i_1,\ldots,i_D}$, such that knowledge of the key will imply access to the resource.

Now suppose that a user $\mathcal{U}$ is authorized to have access to units $X = X_1 \times X_2 \times \cdots \times X_D$, where each X_j is an arbitrary subset of T_j units in dimension j (i.e., unlike the prior work, the intervals in each dimension do not have to be contiguous). With such access rights, $\mathcal{U}$ should receive or should be able to compute the keys $k_{i_1,\ldots,i_D}$, where $i_j \in X_j$ for each j. We denote the private information that $\mathcal{U}$ receives by S_X. Obviously, storing $\prod_{j=1}^{D} |X_j|$ keys at the user end is not always practical, and significantly more efficient solutions are possible. A *multi-dimensional key assignment (MDKA) scheme* assigns keys to the units in a multi-dimensional space and users, so that proper access control is enforced in a correct and efficient manner. Such key generation is assumed to be performed by the resource owner, but once a user is issued the keys, there is no interaction with other entities. More formally, we define a MDKA scheme as follows:

Definition 1. *Let $T = T_1 \times T_2 \times \cdots \times T_D$ define a D-dimensional space. A multi-dimensional key assignment scheme consists of algorithms* $(\mathsf{Setup}, \mathsf{Assign}, \mathsf{Derive})$ *s.t.:*

Setup *is a probabilistic algorithm, which, on input a security parameter 1^κ and D-dimensional grid T, outputs (i) a key $k_{i_1,\ldots,i_D}$ for any $(i_1, \ldots, i_D) \in T$; (ii) secret information* sec *associated with the system; and (iii) public information* pub*. Let* $(K, \mathsf{sec}, \mathsf{pub})$ *denote the output of this algorithm, where K is the set of all keys.*

Assign *is a probabilistic algorithm, which, given specification of access rights $X = X_1 \times \cdots \times X_D \subseteq T$ and secret information* sec*, outputs private information S_X.*

Derive *is a deterministic algorithm, which, on input access rights $X = X_1 \times \cdots \times X_D$, a point $(i_1, \ldots, i_D) \in T$, private information S_X, and public information* pub*, outputs key $k_{i_1,\ldots,i_D}$ if $(i_1, \ldots, i_D) \in X$ and a special failure symbol $\perp$ otherwise. The correctness requirement is such that, for each set of access rights $X \subseteq T$, each point $(i_1, \ldots, i_D) \in X$, each private information S_X, each key $k_{i_1,\ldots,i_D} \in K$, and each public information* pub *that* $\mathsf{Setup}(1^\kappa, T)$ *and* $\mathsf{Assign}(X, \mathsf{sec})$ *can output,* $\Pr[\mathsf{Derive}(X, (i_1, \ldots, i_D), S_X, \mathsf{pub}) = k_{i_1,\ldots,i_D}] = 1$.

Note that we provide a general specification of such a scheme that can work under different assumptions. As mentioned above, in our solution access to the entire public information is not needed, and instead the key derivation algorithm needs access only to the public information for one point in the D-dimensional space, the key of which is being derived. We will denote public information for point $(i_1, \ldots, i_D)$ as $\mathsf{pub}_{i_1,\ldots,i_D}$, and this is what will be needed for Derive. Also, it is possible that in some schemes all

values that the Assign algorithm outputs (i.e., S_X for every $X \subseteq T$) can be produced at the system initialization time (in which case Assign is deterministic), but it is still desired to separate it from Setup.

Security Model. In prior literature on key management schemes, two security goals have been defined [3]: security against *key recovery*, in which an adversary is unable to compute a key to which it should not have access, and security with respect to *key indistinguishability*, which means that an adversary is unable to learn any information about a key to which it should not have access and thus cannot distinguish it from a random string of the same length. The latter is obviously a stronger notion of security. Also, the literature on one-dimensional (i.e., time-based) KA schemes (e.g., [8]) distinguishes between security in the presence of *static adversaries* and security in the presence of *adaptive adversaries*. Then a static adversary is given a specific unit (i.e., a D-dimensional point in our context) to attack and obtains access to all other keys that do not allow it to access the challenge point. An adaptive adversary, on the other hand, obtains oracle access to the Assign algorithm, can query user keys of its choice, choose a challenge unit, and eventually output its response to the challenge.

In [8] it was shown that the security of key assignment schemes against a static adversary is (polynomial-time) equivalent to the security against an adaptive adversary for both security goals (key recovery and key indistinguishability), which on the surface appears to enable us to consider only static adversaries. There is, however, a difference between prior and our specifications of the key assignment algorithm in that we allow it to be probabilistic. This, in particular, means that two users with exactly the same privileges can obtain different secret information that allows them to access the same resources. From the security point of view, this difference is crucial enough that equivalence between security notions in presence of static and adaptive adversaries no longer holds. That is, a static adversary obtains secret information corresponding to a minimal number of users that ensures coverage of keys for all resources except its challenge, while an adaptive adversary can query any number of keys for possibly the same or overlapping access rights. Thus, there can be schemes that are secure if adversary obtains only one set of key material, but insecure when an adversary has access to multiple versions of the secret information for related access rights. Therefore, in the rest of this work we will concentrate on adaptive adversaries only.

Throughout this work, we use notation $a \stackrel{R}{\leftarrow} A$ to mean that a is chosen uniformly at random from the set A. A function $\epsilon(\kappa)$ is *negligible* if for every positive polynomial $p(\cdot)$ and all sufficiently large κ, $\epsilon(\kappa) < \frac{1}{p(\kappa)}$.

Let $\mathcal{A}$ denote an adaptive adversary attacking the security of a MDKA scheme. $\mathcal{A}$ obtains all public information and is given oracle access to Assign algorithm. In the first stage of the attack, $\mathcal{A}$ can query $\mathsf{Assign}(\mathsf{sec}, \cdot)$ and outputs its choice of challenge point $(i_1, \ldots, i_D)$. In the second stage of the attack, $\mathcal{A}$ can further query its oracle and produce its response. Let the set Q denote all queries that $\mathcal{A}$ makes to Assign. $\mathcal{A}$ can query its oracle for any access rights X of its choice with the restriction that the challenge point cannot be contained in X. This in particular does not prevent the adversary from constructing queries that contain all of the i_j's from the challenge across several queries (e.g., querying X_1 and X_2 such that $(i_1, \ldots, i_{D-1}, i'_D) \in X_1$ and $(i'_1, i_2, \ldots, i_D) \in X_2$ is allowed if $i'_D \neq i_D$ and $i'_1 \neq i_1$), which means that the solution must be collusion

resistant. Because the notion of key indistinguishability is strictly stronger than security against key recovery, and it is a widely accepted security model, we concentrate on security with respect to key indistinguishability only. Then after the first stage, $\mathcal{A}$ is given either the real key corresponding to the challenge point or a random value and must correctly guess which one was used. We require that the success probability of $\mathcal{A}$ is negligible in κ. The key indistinguishability experiment is given below.

Experiment $\mathbf{Exp}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T)$
$(K, \mathsf{sec}, \mathsf{pub}) \leftarrow \mathsf{Setup}(1^\kappa, T)$
$((i_1, \ldots, i_D), state) \leftarrow \mathcal{A}_1^{\mathsf{Assign}(\mathsf{sec},\cdot)}(1^\kappa, T, \mathsf{pub})$
$b \overset{R}{\leftarrow} \{0,1\}$
if $b = 0$ then $\alpha \overset{R}{\leftarrow} \{0,1\}^{|k_{i_1,\ldots,i_D}|}$ else $\alpha \leftarrow k_{i_1,\ldots,i_D}$
$b' \leftarrow \mathcal{A}_2^{\mathsf{Assign}(\mathsf{sec},\cdot)}(1^\kappa, T, \mathsf{pub}, (i_1, \ldots, i_D), state, \alpha)$
if $\forall X \in Q, (i_1, \ldots, i_D) \notin X$ and $b = b'$ then return 1 else return 0

Definition 2. *Let $T = T_1 \times \cdots \times T_D$ be a D-dimensional grid of distinct units and $\mathsf{MDKA} = (\mathsf{Setup}, \mathsf{Assign}, \mathsf{Derive})$ be a multi-dimensional key assignment scheme for T and a security parameter κ. Then MDKA is secure with respect to key indistinguishability in the presence of an adaptive adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ with oracle access to $\mathsf{Assign}(\mathsf{sec}, \cdot)$ in both stages of the attack if it satisfies the following properties:*

- Completeness: *A user, who is given private information S_X for access rights to $X = X_1 \times \cdots \times X_D \subseteq T$, is able to compute the access key $k_{i_1,\ldots,i_D}$ for each $(i_1, \ldots, i_D) \in X$ using only her knowledge of S_X and public information pub with probability 1.*
- Soundness: *If we let the experiment $\mathbf{Exp}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}$ be specified as above, the advantage of $\mathcal{A}$ is defined as:*

$$\mathsf{Adv}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T) = \left| \Pr[\mathbf{Exp}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T) = 1] - \tfrac{1}{2} \right|$$

We say that MDKA is sound with respect to key indistinguishability if for each $(i_1, \ldots, i_D) \in T$, for all sufficiently large κ, and every positive polynomial $p(\cdot)$, $\mathsf{Adv}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T) < 1/p(\kappa)$ for each polynomial-time adversary $\mathcal{A}$.

In addition to the security requirements, an efficient MDKA scheme is evaluated by the following criteria:

- The size of the secret data a user must store;
- The amount of computation for generation of an access key for the target resource;
- The amount of information the service provider must maintain.

Number-Theoretic Preliminaries. The notation $\mathbb{G} = \langle g \rangle$ denotes that g generates the group $\mathbb{G}$. Our solution uses groups with pairings, and we review concepts underlying such groups next.

Definition 3 (Bilinear map). *A map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a bilinear map if the following conditions hold:*

- *(Efficient) $\mathbb{G}$ and $\mathbb{G}_T$ are groups of the same prime order q, and there exists an efficient algorithm for computing e.*

– *(Bilinear) For all $g \in \mathbb{G}$, and $a, b \in \mathbb{Z}_q$, $e(g^a, g^b) = e(g, g)^{ab}$.*
– *(Non-degenerate) If g generates $\mathbb{G}$, then $e(g, g)$ generates $\mathbb{G}_T$.*

Throughout this work, we assume that there is a setup algorithm Set that, on input a security parameter 1^κ, outputs the setup for group $\mathbb{G} = \langle g \rangle$ of prime order q that have a bilinear map e, and $h = e(g, g)$ generates $\mathbb{G}_T$ (which also has order q). That is, $(q, \mathbb{G}, \mathbb{G}_T, e, g, h) \leftarrow Set(1^\kappa)$.

The security of our scheme relies on Decision Linear Diffie-Hellman assumption (DLIN). It was introduced in [10] and is currently widely used; we review it next.

Definition 4 (DLIN). *The Decision Linear problem is, given generator g of $\mathbb{G}$, g^a, g^b, g^{ac}, g^{bd}, and Z, where $a, b, c, d \in \mathbb{Z}_q$ and $Z \in \mathbb{G}$, output 1 if $Z = g^{c+d}$ and 0 otherwise. We say that the Decision Linear assumption holds in $\mathbb{G}$ if any probabilistic polynomial time (in κ) adversary $\mathcal{A}$ has at most negligible probability in solving the Decision Linear problem. More precisely,*

$$\begin{aligned}\mathsf{Adv}_{\mathsf{DLIN},\mathcal{A}}(1^\kappa) = |\Pr[\mathcal{A}(\mathbb{G}, q, g, g^a, g^b, g^{ac}, g^{bd}, g^{c+d}) = 1] - \\ -\Pr[\mathcal{A}(\mathbb{G}, q, g, g^a, g^b, g^{ac}, g^{bd}, g^R) = 1]| \leq \epsilon(\kappa)\end{aligned}$$

for some negligible function $\epsilon(\cdot)$.

4 Description of the Scheme

Overview of the Scheme. Our solution was inspired by work on multi-dimensional range queries [19], where a secret was used to tie multiple dimensions to achieve collusion resilience in a different context. That high-level idea led us to develop a new scheme which is more balanced than all existing key management solutions and improves their performance. Furthermore, our approach supports a richer set of access rights than prior key management work.

At a high level, in our scheme each point j in the ith dimension (for $1 \leq i \leq D$ and $1 \leq j \leq T_i$) is assigned a secret $s_{i,j}$. There is also a system-wide secret w. When a user subscribes to the resources in $X = X_1 \times \cdots \times X_D$, she is issued keys, or private information S_X, that are a function of both $s_{i,j}$'s in her access rights X and w. In particular, w is first split into D random shares w_i such that $\sum_{i=1}^{D} w_i = w$. Then for each point j in the ith dimension of user's subscription (i.e., $j \in X_i$ for $1 \leq i \leq D$), the user obtains a key $k_{i,j}$ computed u sing $s_{i,j}$ and w_i.

When a user receives a broadcast and wants to compute a key associated with a point $(i_1, \ldots, i_D)$, she will be able to derive the encryption key for that point only if $i_j \in X_j$ for each $1 \leq j \leq D$. To compute the encryption key, the user retrieves the key k_{j,i_j} from S_X corresponding to each coordinate i_j of the point $(i_1, \ldots, i_D)$ in dimension j. The point $(i_1, \ldots, i_D)$ will also have publicly available information consisting of $D+1$ values (which is included in the broadcast). The user combines elements of that public data with her keys k_{j,i_j} dimension-wise to compute the necessary encryption key.

Detailed Description. We now present a complete description of the scheme. Security analysis is given in Section 5, and performance analysis in Section 7.

Setup : Run $(q, \mathbb{G}, \mathbb{G}_T, e, g, h) \leftarrow Set(1^\kappa)$ to generate a group with pairings. Choose the master secret $w \stackrel{R}{\leftarrow} \mathbb{Z}_q$. For each dimension i, for each unit j in dimension i, choose its secret $s_{i,j} \stackrel{R}{\leftarrow} \mathbb{Z}_q$. For each D-dimensional point with coordinates $(i_1, i_2, \ldots, i_D) \in T$, generate public information by choosing $r_{(i_1,\ldots,i_D)} \stackrel{R}{\leftarrow} \mathbb{Z}_q$ and setting $\mathsf{pub}_{i_1,\ldots,i_D} = (g^{r_{(i_1,\ldots,i_D)}}, g^{r_{(i_1,\ldots,i_D)} \cdot s_{1,i_1}}, \ldots, g^{r_{(i_1,\ldots,i_D)} \cdot s_{D,i_D}})$. The key for point $(i_1, i_2, \ldots, i_D)$ is $e(g,g)^{r_{(i_1,\ldots,i_D)} w}$ for the value of $r_{(i_1,\ldots,i_D)}$ used in producing the public data.

Assign : Suppose user $\mathcal{U}$ is entitled to access privileges to a D-dimensional structure $X = X_1 \times X_2 \times \cdots \times X_D$, where for each dimension i, $X_i \subseteq 2^{T_i}$ (i.e., X_i can be an arbitrary subset of T_i items). First, randomly choose D random values $w_1, \ldots, w_D$ from $\mathbb{Z}_q$ subject to the constraint $\sum_{i=1}^{D} w_i \bmod q = w$. For each $i = 1, \ldots, D$, for each $j \in X_i$, randomly choose $t \stackrel{R}{\leftarrow} \mathbb{Z}_q$ and add $k_{i,j} = (g^t, g^{w_i + t \cdot s_{i,j}})$ to the user's S_X.

KeyDer : A user who is entitled to access a D-dimensional point with coordinates $(i_1, \ldots, i_D)$ first retrieves the key associated with each coordinate i_j from her private information S_X. Let $k_{j,i_j} = (g^{t_j}, g^{w_j + t_j \cdot s_{j,i_j}})$ denote such a key. Next, the user retrieves the public information associated with the point $\mathsf{pub}_{i_1,\ldots,i_D} = (g^{r_{(i_1,\ldots,i_D)}}, g^{r_{(i_1,\ldots,i_D)} \cdot s_{1,i_1}}, \ldots, g^{r_{(i_1,\ldots,i_D)} \cdot s_{D,i_D}})$ from the broadcasted content and derives the encryption key as:

$$\prod_{j=1}^{D} e(k_{j,i_j}[2], \mathsf{pub}_{i_1,\ldots,i_D}[1]) e((k_{j,i_j}[1])^{-1}, \mathsf{pub}_{i_1,\ldots,i_D}[j+1])$$

Here $u[i]$ denotes the ith value of tuple u.

It is clear from the above that a system consisting of $\prod_{i=1}^{D} T_i$ points in the D-dimensional space will support $\prod_{i=1}^{D} 2^{T_i}$ types of access privileges. While there is public information associated with each point in T, in Section 6 we show that in practice the service owner needs to have only $O(1)$ storage to maintain the operation of the system.

5 Security Analysis

In this section we show that our scheme satisfies both the completeness and soundness requirements. Efficiency of our solution is evaluated in Section 7.

Theorem 1. *The multi-dimensional key assignment scheme* $\mathsf{MKDA} = (\mathsf{Setup}, \mathsf{Assign}, \mathsf{KeyDir})$ *presented above is complete.*

It is not difficult to show that the result of computation performed at key derivation time for a grid point always equals to the encryption key generated for that point at the setup time. We omit the details due to space considerations.

Theorem 2. *Assuming that the Decision Linear assumption holds, the multi-dimensional key assignment scheme* $\mathsf{MDKA} = (\mathsf{Setup}, \mathsf{Assign}, \mathsf{KeyDir})$ *presented above achieves key indistinguishability in the presence of adaptive adversaries.*

Proof. Suppose there is a PPT adversary $\mathcal{A}$ such that $\mathsf{Adv}^{\mathrm{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T) > 1/p(\kappa)$ for some polynomial p. We will show that there exists a PPT adversary $\mathcal{B}$ with black box access to $\mathcal{A}$ that solves the decision linear problem with non-negligible probability.

According to the definition, $\mathcal{B}$ is given $(q, \mathbb{G}, \mathbb{G}_T, e, g, h), g^a$, g^b, g^{ac}, g^{bd}, and Z, and need to decide whether $Z = g^{c+d}$. In our case, $\mathcal{B}$ will be given Z of the form g^{c+d} or g^{R+d} for some $R \in \mathbb{Z}_q$ (where each element of the group can be written as g^{R+d} for some R) and needs to correctly decide whether it was g^{c+d}.

Our algorithm $\mathcal{B}$ first chooses a random point $(\hat{i}_1, \ldots, \hat{i}_D) \in T$. Essentially, $\mathcal{B}$ is guessing the point which $\mathcal{A}$ will use as its challenge. $\mathcal{B}$ then interacts with $\mathcal{A}$ as follows:

Setup: $\mathcal{B}$ performs system setup as follows:

1. It sets the parameters using $(q, \mathbb{G}, \mathbb{G}_T, e, g, h)$.
2. To generate secret information for the cells of the grid, for each dimension $j \in [1, D]$ and each element $t \in [1, T_j]$ in dimension j, $\mathcal{B}$ chooses a random element $q_{j,t} \stackrel{R}{\leftarrow} \mathbb{Z}_q$. $\mathcal{B}$ stores these q values. To finish the setup of secret information sec, we implicitly set $s_{j,t} = q_{j,t}$ when $\hat{i}_j = t$ (i.e., it is part of the challenge) and to $s_{j,t} = q_{j,t} + d$ otherwise. Note that in the latter case $\mathcal{B}$ does not know $s_{j,t}$.
3. To generate the public information for each point $(i_1, \ldots, i_D)$, there are two cases:
   - $(i_1, \ldots, i_D) = (\hat{i}_1, \ldots, \hat{i}_D)$: In this case, we use g^a from $\mathcal{B}$'s challenge to set $\mathsf{pub}_{i_1,\ldots,i_D} = (g^a, (g^a)^{q_{1,i_1}}, \ldots, (g^a)^{q_{D,i_D}})$. This means that $r_{(i_1,\ldots,i_D)} = a$. Notice that $\mathsf{pub}_{i_1,\ldots,i_D} = (g^{r_{(i_1,\ldots,i_D)}}, g^{r_{(i_1,\ldots,i_D)} \cdot s_{1,i_1}}, \ldots, g^{r_{(i_1,\ldots,i_D)} \cdot s_{D,i_D}})$, which is the same as in the real protocol.
   - $(i_1, \ldots, i_D) \neq (\hat{i}_1, \ldots, \hat{i}_D)$: Choose random $u_{(i_1,\ldots,i_D)} \stackrel{R}{\leftarrow} \mathbb{Z}_q$ and use g^b, g^{bd} from $\mathcal{B}$'s challenge to set $\mathsf{pub}_{i_1,\ldots,i_D} = ((g^b)^{u_{(i_1,\ldots,i_D)}}, R_1, \ldots, R_D)$, where

$$R_j = \begin{cases} (g^b)^{u_{(i_1,\ldots,i_D)} \cdot q_{j,i_j}} & \text{if } i_j = \hat{i}_j \\ (g^{bd})^{u_{(i_1,\ldots,i_D)}} (g^b)^{u_{(i_1,\ldots,i_D)} \cdot q_{j,i_j}} & \text{otherwise.} \end{cases}$$

     This means that $\mathcal{B}$ sets $r_{(i_1,\ldots,i_D)} = b \cdot u_{(i_1,\ldots,i_D)}$ and $R_j = g^{r_{(i_1,\ldots,i_D)} \cdot s_{j,i_j}}$, where $s_{j,i_j} = q_{j,i_j}$ if $i_j = \hat{i_j}$ and $s_{j,i_j} = q_{j,i_j} + d$ otherwise (which is consistent with the way secret information was setup). The above guarantees that these tuples are distributed identically to when $\mathcal{A}$ engages in the real protocol.

Assign **queries**: When $\mathcal{A}$ asks for a query $X = X_1 \times X_2 \times \cdots \times X_D$, $\mathcal{B}$ responds as:

1. If $\hat{i}_j \in X_j$ for each $1 \leq j \leq D$ (i.e., X contains the challenge), $\mathcal{B}$ outputs $FAIL$.
2. Otherwise, $\mathcal{B}$ chooses m to be an index such that $\hat{i}_m \notin X_m$ (there must be at least one such index). Next, $\mathcal{B}$ chooses and stores random values $w_1, \ldots, w_{m-1}, w_{m+1}, \ldots, w_D$ from $\mathbb{Z}_q$. Let $w' = \sum_{i \in [1,D], i \neq m} w_i$. $\mathcal{B}$ creates the key material by setting the key information for dimension j and position $t \in X_j$ as follows:
   - If $j = m$, then choose $\ell \stackrel{R}{\leftarrow} \mathbb{Z}_q$ and use g^b, g^{bd}, Z from $\mathcal{B}$'s challenge to compute and return $k_{j,t} = ((g^b)^\ell g,\ Z(g^{bd})^\ell g^{-w'} (g^b)^{\ell q_{j,t}} g^{q_{j,t}}) =$
     $= (g^{b\ell+1}, Z g^{bd\ell - w' + b\ell q_{j,t} + q_{j,t}}) = (g^{b\ell+1}, Z g^{b\ell(d+q_{j,t}) + q_{j,t} - w'})$.
     Note that, because in this case $s_{j,t} = q_{j,t} + d$, when $Z = g^{c+d}$, this tuple is $(g^{b\ell+1}, g^{c - w' + (b\ell+1)(d+q_{j,t})}) = (g^{b\ell+1}, g^{c - w' + (b\ell+1) s_{j,t}})$. Otherwise, when $Z = g^{R+d}$, it is $(g^{b\ell+1}, g^{R - w' + (b\ell+1) s_{j,t}})$.
   - If $j \neq m$ and $\hat{i}_j \neq t$, first choose $\ell \stackrel{R}{\leftarrow} \mathbb{Z}_q$, then compute and return $k_{j,t} = ((g^b)^\ell, g^{w_j} (g^b)^{\ell q_{j,t}} (g^{bd})^\ell) = (g^{b\ell}, g^{w_j + b\ell(q_{j,t}+d)}) = (g^{b\ell}, g^{w_j + b\ell s_{j,t}})$, where $s_{j,t} = q_{j,t} + d$ as required.

– If $j \neq m$ and $\hat{i}_j = t$, choose $\ell \xleftarrow{R} \mathbb{Z}_q$ and return $k_{j,t} = (g^\ell, g^{w_j} g^{\ell q_{j,t}}) = (g^\ell, g^{w_j + \ell s_{j,t}})$, where now $s_{j,t} = q_{j,t}$ as previously set.

Notice that the keys are consistent with those generated by a real challenger. In particular, for each dimension j and each element $t \in X_j$, $k_{j,t}$ is of the form $(g^r, g^{w_i + r s_{j,t}})$, where r takes the value of $b\ell + 1$, $b\ell$, or ℓ depending on the case, and $w = \sum_{i=1}^{D} w_i$. Note that we have implicitly defined w using randomly chosen w_i for all but one dimension and Z. More specifically, if $Z = g^{c+d}$, then $w = c$ and if $Z = g^{R+d}$, then $w = R$. In all cases $\mathcal{B}$ does not know w. Furthermore, this key assignment implies that applying the key derivation procedure to the key for $(\hat{i}_1, \ldots, \hat{i}_D)$ and $\mathsf{pub}_{\hat{i}_1,\ldots,\hat{i}_D}$ results in the encryption key $e(g,g)^{ac}$ when $Z = g^{c+d}$ and $e(g,g)^{aR}$ when $Z = g^{R+d}$.

Challenge: When $\mathcal{A}$ issues a challenge for $(\bar{i}_1, \ldots, \bar{i}_D)$, if $(\bar{i}_1, \ldots, \bar{i}_D) \neq (\hat{i}_1, \ldots, \hat{i}_D)$, $\mathcal{B}$ outputs $FAIL$. Otherwise, $\mathcal{B}$ returns $e(g^{ac}, g) = e(g,g)^{ac}$. If $Z = g^{c+d}$, then this is the correct key, but if $Z = g^{R+d}$, then this is a independent key from the real one specified by the above parameters.

More Assign **queries**: Same as before.

Output: Eventually, $\mathcal{A}$ outputs a bit b' and $\mathcal{B}$ returns b'.

Suppose $\mathcal{B}$ does not output $FAIL$. Then if $Z = g^{c+d}$, $\mathcal{A}$ has been given the correct key for the challenge point, and if $Z = g^{R+d}$, $\mathcal{A}$ is given a random key. This means that $\mathcal{A}$'s view is the same as in $\mathbf{Exp}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T)$. Furthermore, because $\mathcal{B}$ simply outputs what $\mathcal{A}$'s outputs, if $\mathcal{A}$ can distinguish keys with non-negligible probability, $\mathcal{B}$ will also be able to solve the decision linear problem with non-negligible probability. We next give a more detailed analysis to tie the advantage of $\mathcal{A}$ in experiment $\mathbf{Exp}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T)$ with the advantage of $\mathcal{B}$ in solving the decision linear problem.

First observe that:

$$\Pr[\mathbf{Exp}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T) = 1] = \Pr[\mathbf{Exp}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T) = 1 \wedge \mathsf{ProperQueries}],$$

where the event $\mathsf{ProperQueries}$ means that $\mathcal{A}$ did not query X containing the challenge point during any of its calls to Assign. This equality is true because the experiment always outputs 0 when $\mathcal{A}$ violates this querying constraint. We also use $\mathsf{GoodGuess}$ to denote the event when $\mathcal{A}$ guesses the bit b in the experiment correctly (i.e., when $b = b'$). This in particular implies that $\Pr[\mathbf{Exp}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T) = 1] = \Pr[\mathsf{GoodGuess} \wedge \mathsf{ProperQueries}]$. Next, we have:

$$\begin{aligned}
\mathsf{Adv}_{\mathsf{DLIN},\mathcal{B}}(1^\kappa) &= \big|\Pr[\mathcal{B}(\mathbb{G}, q, g, g^a, g^b, g^{ac}, g^{bd}, g^{c+d}) = 1] - && (1)\\
&\quad - \Pr[\mathcal{B}(\mathbb{G}, q, g, g^a, g^b, g^{ac}, g^{bd}, g^R) = 1]\big| && (2)\\
&= \big|\Pr[\mathsf{GoodGuess} \wedge \overline{\mathsf{Fail}}] - \Pr[\overline{\mathsf{GoodGuess}} \wedge \overline{\mathsf{Fail}}]\big| && (3)
\end{aligned}$$

where Fail denotes the event that $\mathcal{B}$ outputs $FAIL$ as a result of interaction with $\mathcal{A}$. From the description of the interaction, we know that $\mathcal{B}$ outputs $FAIL$ when (i) $\mathcal{B}$ does not guess the challenge correctly or (ii) when $\mathcal{A}$ attempts to query a key for privileges that contain the point chosen to be the challenge. We formalize this as $\Pr[\overline{\mathsf{Fail}}] = \Pr[\overline{\mathsf{WrongChallenge}} \wedge \mathsf{ProperQueries}]$. Substituting this into equation (3), we obtain:

$$\mathsf{Adv}_{\mathsf{DLIN},\mathcal{B}}(1^\kappa) = \big|\Pr[\mathsf{GoodGuess} \wedge \overline{\mathsf{WrongChallenge}} \wedge \mathsf{ProperQueries}] -$$

$$
\begin{aligned}
&\quad -\Pr[\overline{\mathsf{GoodGuess}} \wedge \overline{\mathsf{WrongChallenge}} \wedge \mathsf{ProperQueries}]\big| \\
&= \big|\Pr[\overline{\mathsf{WrongChallenge}} \mid \mathsf{GoodGuess} \wedge \mathsf{ProperQueries}] \times \\
&\quad \times \Pr[\mathsf{GoodGuess} \wedge \mathsf{ProperQueries}] - \\
&\quad -\Pr[\overline{\mathsf{WrongChallenge}} \mid \overline{\mathsf{GoodGuess}} \wedge \mathsf{ProperQueries}] \times \\
&\quad \times \Pr[\overline{\mathsf{GoodGuess}} \wedge \mathsf{ProperQueries}]\big|
\end{aligned}
$$

Now because $\mathcal{B}$ chooses its challenge point uniformly at random regardless of $\mathcal{A}$'s behavior, we rewrite the above as:

$$
\begin{aligned}
\mathsf{Adv}_{\mathsf{DLIN},\mathcal{B}}(1^\kappa) &= \big|\Pr[\overline{\mathsf{WrongChallenge}}]\Pr[\mathsf{GoodGuess} \wedge \mathsf{ProperQueries}] - \\
&\quad -\Pr[\overline{\mathsf{WrongChallenge}}]\Pr[\overline{\mathsf{GoodGuess}} \wedge \mathsf{ProperQueries}]\big| \\
&= \left|\frac{1}{T}\Pr[\mathbf{Exp}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T) = 1] - \right. \\
&\quad \left. -\frac{1}{T}\left(1 - \Pr[\mathbf{Exp}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T) = 1] - \Pr[\overline{\mathsf{ProperQueries}}]\right)\right|
\end{aligned}
$$

since $\Pr[\overline{\mathsf{WrongChallenge}}] = \frac{1}{T}$ and

$$
\begin{aligned}
&1 = \Pr[\mathsf{ProperQueries}] + \Pr[\overline{\mathsf{ProperQueries}}] = \\
&= \Pr[\mathsf{GoodGuess} \wedge \mathsf{ProperQueries}] + \Pr[\overline{\mathsf{GoodGuess}} \wedge \mathsf{ProperQueries}] + \Pr[\overline{\mathsf{ProperQueries}}] = \\
&= \Pr[\mathbf{Exp}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T) = 1] + \Pr[\overline{\mathsf{GoodGuess}} \wedge \mathsf{ProperQueries}] + \Pr[\overline{\mathsf{ProperQueries}}]
\end{aligned}
$$

Finally, we obtain

$$
\begin{aligned}
\mathsf{Adv}_{\mathsf{DLIN},\mathcal{B}}(1^\kappa) &= \frac{1}{T}\left|2\Pr[\mathbf{Exp}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T) = 1] - 1 + \Pr[\overline{\mathsf{ProperQueries}}]\right| \\
&\geq \frac{1}{T}\left|2\Pr[\mathbf{Exp}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T) = 1] - 1\right| \\
&= \frac{2}{T}\left|\Pr[\mathbf{Exp}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T) = 1] - \frac{1}{2}\right| = \frac{2}{T}\mathsf{Adv}^{\text{key-ind}}_{\mathsf{MDKA},\mathcal{A}}(1^\kappa, T)
\end{aligned}
$$

This means that if $\mathcal{A}$ succeeds in breaking the security of the MDKA scheme with non-negligible probability δ, $\mathcal{B}$ succeeds in breaking the decisional linear problem with non-negligible probability which is at least $2\delta/T$. □

The above reduction relates the success probabilities of algorithms $\mathcal{A}$ and $\mathcal{B}$ using a factor of $2/T$. This means that it is desirable to set the security parameter of the scheme to be $\kappa + \log(T) - 1$, where κ is the security parameter necessary to ensure the difficulty of solving the decision linear problem. This is likely to increase κ by a few dozen bits (see, e.g., Section 7 for an example application).

Note that our result is consistent with best practices in the literature (e.g., [8]), where security against adaptive adversaries is desired (i.e., the simulator is forced to guess the challenge point). Furthermore, related work on range queries and IBE-based schemes have security proofs in a weaker, so-called selective ID model, where the adversary commits to the challenge point prior to system setup. Under those circumstances, we would achieve a tight reduction with no efficiency loss.

6 Extensions

Reducing Public Storage. In the system the way it was described, the public storage at the server is $O(TD)$. As T could be large, this amount of storage may be problematic. Furthermore, the setup algorithm requires this many modular exponentiations, which is also a bottleneck. We modify the scheme in order to reduce the storage to $O(1)$. The crux of this idea is that since we send the public information to the user on demand, we do not need to have all of the information at once. Furthermore, this information can be derived as it is needed. More specifically, let $F_1 : [1, D] \times T_{\max} \times \{0,1\}^\kappa \to \mathbb{Z}_q$ and $F_2 : T \times \{0,1\}^\kappa \to \mathbb{Z}_q$ be pseudorandom functions, where $T_{\max}$ is the maximum of $T_1, \ldots, T_D$. We make the following changes to the MDKA scheme:

Setup: In this case the public information is now just $(q, \mathbb{G}, \mathbb{G}_T, e, g, h)$. We still chooses the master secret $w \xleftarrow{R} \mathbb{Z}_q$ along with two PRF keys $k_1 \xleftarrow{R} \{0,1\}^\kappa$ and $k_2 \xleftarrow{R} \{0,1\}^\kappa$. The secret information is then (w, k_1, k_2). Implicitly we are setting the secret parameters to be $s_{i,j} = F_1(i, j, k_1)$ and $r_{(i_1,\ldots,i_D)} = F_2((i_1, \ldots, i_D), k_2)$.

Assign and KeyDer: When we need to compute a user's key or public information, we simply compute its values using F_1 and F_2.

One-Time Keys. One concern with key management solutions is that users can distribute access keys to unauthorized parties. This would allow anyone to access the content for free. There are two types of such revelations possible for our system: (i) the user can publish its private S_X or (ii) she can derive the key for a specific cell $(i_1, \ldots, i_D)$ and publish it (i.e., publish $e(g,g)^{r_{(i_1,\ldots,i_D)} \cdot w}$). Of these two types of distributions, the latter is worse, because it reveals no information about the offending party except the ability to access point $(i_1, \ldots, i_D)$, whereas the first type reveals significantly more information about that party, i.e, the complete specification of the access rights. Fortunately, the latter, more damaging attack can be mitigated as follows: The value $r_{(i_1,\ldots,i_D)}$ can be changed each time that cell is used. That is, since we are sending $\mathsf{pub}_{i_1,\ldots,i_D}$ to the users along with the ciphertext, the protocol can simply choose a new values of $r_{(i_1,\ldots,i_D)}$ each time. Thus, the key $e(g,g)^{r_{(i_1,\ldots,i_D)} \cdot w}$ is useful only for the current message, and will not be useful for other messages.

7 Performance

The complexity of our MDKA scheme is as follows:

1. *Size of* pub: This is $O(1)$ as the only values that need to be stored are $(q, \mathbb{G}, \mathbb{G}_T, e, g, h)$.
2. *Size of* sec: This has size $O(1)$ as all that is stored is (w, k_1, k_2).
3. *Size of user key:* A user with access to $X_1 \times \cdots \times X_D$ obtains $\sum_{i=1}^{D} |X_i|$ pairs of values as its private keys and thus maintains $O(\sum_{i=1}^{D} |X_i|)$ values.
4. *Size of an encryption:* To be able to decrypt, the user needs to have the public information associated with the access point, which has size $O(D)$.
5. *Cost to assign key:* This requires $O(\sum_{i=1}^{D} |X_i|)$ work.
6. *Cost to send broadcast to user:* The user will need to receive the public information, which will require $O(D)$ operations from the sender.

7. *Cost to derive key:* This requires $O(D)$ operations.

To demonstrate the applicability of our approach to practical systems, we consider a content streaming application. We give a small example and discuss the performance of our scheme. Suppose that a content streaming system has the following dimensions:

(i) the specific content (i.e., show) being accessed inside the range $[1, 2^{16}]$,
(ii) quality of programming inside the range $[1, 8]$,
(iii) time of access inside the range $[1, 7760]$ (i.e., once for every hour of a year),
(iv) x-coordinate of location of access ranging in the range $[1, 1024]$, and
(v) y-coordinate of location of access ranging in the range $[1, 1024]$.

The motivation for the locations is that the service provider desires to only let the user see the content in certain locations for DRM purposes and location-based content (such as local weather forecast). Consider a user that subscribes to 100 shows, two quality markers (one poor quality for a mobile device and one high quality for home), for access from 6-10PM daily, in a 10 by 10 region. Using our solution, this user would need to store $100 + 2 + 1460 + 10 + 10 = 1582$ keys, which is clearly practical. Furthermore, to derive a specific key, a user would have to perform 10 pairing operations. According to [16], each of these operations take about 11ms on a 1 GHz Pentium III, and thus key derivation would require about 110ms, which is also clearly practical.

We now consider the shortcoming of the solution presented in [20] for this particular problem. First, this scheme provides a weaker notion of security (key recovery instead of key indistinguishability). The main problem, however, is that this scheme does not support arbitrary intervals. This means that the content and time blocks must be separated and multiple sets of keys must be given to the user. That is, for each of the 100 shows and for each day, the user must have $2^3(\frac{3+10+10+13}{4}) = 72$ keys, and therefore store $72 \cdot 365 \cdot 100$ or about 2.6 million keys. Clearly, as the subscription becomes more complex, this will result in the user storing too many keys.

8 Conclusions

In this work we treat the problem of key assignment in multi-dimensional space for subscription-based broadcast and location-based services. In particular, each dimension corresponds to an attribute (such as latitude, longitude, time, or any other attribute) which is partitioned into a number of units, comprising a D-dimensional grid. All resources associated with a point in this grid are assigned a cryptographic key and distributed in encrypted form. A subscriber joining the system obtains access to certain resources specified as a subset of points in each dimension. She is issued key material that allows her to derive cryptographic keys for all D-dimensional points in the subscription privileges. We give a new scheme for key assignment and management with characteristics that favorably compare with existing schemes. In particular, the user acquires overhead only linear in the number of dimensions, is not required to access external data in addition to broadcast content, and can be issued more flexible access privileges than in other schemes. Our solution is provably secure under the standard Decision Linear assumption.

References

1. Garmin, 2009. http://www.garmin.com.
2. LOC-AID, 2009. http://www.loc-aid.net.
3. M. Atallah, M. Blanton, N. Fazio, and K. Frikken. Dynamic and efficient key management for access hierarchies. *ACM Transactions on Information and System Security (TISSEC)*, 12(3):1–43, January 2009.
4. M. Atallah, M. Blanton, and K. Frikken. Key management for non-tree access hierarchies. In *ACM Symposium on Access Control Models and Technologies*, pages 11–18, 2006.
5. M. Atallah, M. Blanton, and K. Frikken. Efficient techniques for realizing geo-spatial access control. In *ACM Symposium on Information, Computer and Communications Security (ASIACCS'07)*, pages 82–92, 2007.
6. M. Atallah, M. Blanton, and K. Frikken. Incorporating temporal capabilities in existing key management schemes. In *ESORICS*, pages 515–530, 2007.
7. M. Atallah, K. Frikken, and M. Blanton. Dynamic and efficient key management for access hierarchies. In *ACM CCS*, pages 190–201, 2005.
8. G. Ateniese, A. De Santis, A. Ferrara, and B. Masucci. Provably-secure time-bound hierarchical key assignment schemes. In *ACM CCS*, pages 288–297, 2006.
9. D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In *Advances in Cryptology – EUROCRYPT*, pages 440–456, 2005.
10. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Advances in Cryptology – CRYPTO'04*, volume 3152 of *LNCS*, pages 41–55, 2004.
11. D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In *Theory of Cryptography Conference (TCC'07)*, volume 4392 of *LNCS*, pages 535–554, 2006.
12. A. De Santis, A. Ferrara, and B. Masucci. New constructions for provably-secure time-bound hierarchical key assignment schemes. In *ACM Symposium on Access Control Models and Technologies (SACMAT'07)*, pages 133–138, 2007.
13. H. Harney and C. Muckenhirn. Group key management protocol (GKMP) architecture. IETF RFC 2094, 1997. http://www.rfc-archive.org/getrfc.php?rfc=2094.
14. H. Harney and C. Muckenhirn. Group key management protocol (GKMP) specification. IETF RFC 2093, 1997. http://www.rfc-archive.org/getrfc.php?rfc=2093.
15. H. Huang and C. Chang. A new cryptographic key assignment scheme with time-constraint access control in a hierarchy. *Computer Standards & Interfaces*, 26:159–166, 2004.
16. Ben Lynn. The pairing-based cryptography (pbc) library. http://crypto.stanford.edu/pbc.
17. C. Patterson, R. Muntz, and C. Pancake. Challenges in location-aware computing. *IEEE Pervasive Computing*, 2(2):80–89, 2003.
18. A. Perrig, D. Song, and J. Tygar. ELK: A new protocol for efficient large group key distribution. In *IEEE Symposium on Security and Privacy*, pages 247–262, 2001.
19. E. Shi, J. Bethencourt, H. Chan, D. Song, and A. Perrig. Multi-dimensional range query over encrypted data. In *IEEE Security and Privacy Symposium*, pages 350–364, 2007.
20. M. Srivatsa, A Iyengar, J. Yin, and L. Liu. Access control in location-based broadcast services. In *IEEE INFOCOM*, pages 256–260, 2008.
21. W. Tzeng. A time-bound cryptographic key assignment scheme for access control in a hierarchy. *IEEE Transactions on Knowledge and Data Engineering*, 14(1):182–188, 2002.
22. S.-Y. Wang and C.-S. Laih. Merging: An efficient solution for a time-bound hierarchical key assignment scheme. *IEEE Transactions on Dependable and Secure Computing*, 3(1):91–100, 2006.
23. C. Wong, M. Gouda, and S. Lam. Secure group communication using key graphs. In *ACM SIGCOMM*, pages 68–79, 1998.
24. H. Yuan and M. Atallah. Efficient and secure distribution of massive geo-spatial data. In *ACM GIS*, pages 440–443, 2009.