Dynamic Inference Control

Jessica Staddon
Palo Alto Research Center
3333 Coyote Hill Rd.
Palo Alto, CA 94304, USA
staddon@parc.com

ABSTRACT

An inference problem exists in a multilevel database if knowledge of some objects in the database allows information with a higher security level to be inferred. Many such inferences may be prevented prior to any query processing by raising the security level of some of the objects, however this inevitably impedes information access, as a user with low authorization who queries just one of the objects with raised security must seek clearance even when not in danger of making the inference. More flexible access control is possible when inferences are prevented during query processing, however this practice can result in slow query response times. We demonstrate that access control can be made sufficiently dynamic to ensure easy access to the information users are entitled to, while retaining fast query processing. Our inference control schemes provide collusion resistance and have a query processing time that depends only on the length of the inference channels (not on the length of user query histories). In addition, our schemes provide a property we call *crowd control* that goes beyond collusion resistance to ensure that if a large number of users have queried all but one of the objects in an inference channel, then no one will be able to query the remaining object regardless of the level of collusion resistance provided by the scheme.

1. INTRODUCTION

In a multilevel database an inference problem exists if users are able to infer sensitive information from a sequence of queries that each have a low security classification (i.e. are not sensitive). For example, any user may be able to query a database to retrieve a list of ship names and the ports at which they are docked. In addition, the knowledge of which ports are being used to load ships with weapons may have a low security classification. Yet, these two queries together constitute an *inference channel*, because if both are made then it's possible to infer exactly which ships are carrying weapons, and this may be sensitive.

When protecting against inferences, there is an inherent trade-off between the granularity of the access control and the query processing time. The approach to inference control proposed in [17; 21] requires essentially no query processing. In [17; 21], the security levels of specific objects in the database are raised in order to prevent a user with low security clearance from completing enough queries to be able to make an undesired inference. By ensuring that at least one object in each inference channel requires high clearance, low-clearance users are prevented from making inferences. However, a user who only wants to query one particular object whose security level has been raised will be unable to do so without receiving additional authorization, even though the information they seek may be completely innocuous on its own. Hence, because access controls are predetermined, this approach may impede access to information unnecessarily .

Another approach to inference control is to determine at query time whether a query can be safely answered. This can be done, for example, by maintaining user query histories. When a user makes a query it is checked against the user's query history and all known inference channels, before granting access to the results of the query. More sophisticated methods of query-time inference control use inference engines to determine access [9]. However, since query histories can be quite long, this approach can result in slow query processing.

We introduce a new approach to inference control that allows for fast query processing while enabling fine-grained access control, and thus, flexible information access. In our approach, access-enabling tokens are associated with objects in the database, and users are allocated keys that they use to generate the necessary tokens. Once a token is used to query an object the key it was derived from cannot be used to query any other object in the inference channel. This is implemented by deleting (or, revoking) the tokens generated with this key from other objects in the channel. Hence, query processing depends on the length of the channel rather than the ever-growing user query histories. In addition, because initially the same tokens are associated with each object, our approach allows for flexible information access. A user can access any objects in the inference channel provided doing so will not enable the user to make the undesired inference, even through collusion with other users.

Our approach to inference control is inspired by cryptographic revocation techniques for large groups of users. The motivating intuition behind the use of these techniques is that group dynamics play an essential role in ensuring inference control: it is not enough to only consider the user requesting the object when deciding whether or not to grant access, instead all of the users of the database and all of the queries they've made, should somehow be taken into account. The difficulty comes in finding a way to do this without relying on the time-consuming processing of user query histories. As an example, one might imagine solving the problem by associating counters with objects in the channel, and cutting off access to the channel when the counters get too high. However, the counters reflect x queries by 1 user and 1 query by each of x users in the same way, and this doesn't sufficiently capture the access dynamics. By leveraging ideas from group communications we are able to provide some separation between these cases and an automated mechanism for updating the access controls that is far more likely to be affected by large-scale querying by a few users, than scattered queries by many users. More precisely, our schemes provide collusion resistance and a desirable new property that we call *crowd control*. Crowd control ensures that if a large number of users have queried all but one of the objects in the channel then no one will be able to query the remainder of the channel even if they have never queried the database before.

OVERVIEW. In Section 1.1 we discuss related work and in Section 2 we provide the necessary background definitions and notation. In Section 3 we provide our two inference control schemes and in Section 4 we discuss their attributes and highlight areas for future work.

1.1 Related Work

Our approach to inference control relies on the identification of inference channels in an initial pre-query processing step. Of course, it is impossible to identify all inference channels [24] but previous work has demonstrated significant success in identifying inferences both from the database schema [6; 17; 2; 26] and the data itself [5; 8; 28]. Our inference control techniques work with either approach, however we note that if the latter approach is used then it should be re-run periodically to accommodate changes in the data and this *could* lead to a need to distribute additional keys to users (if, for example, an inference channel of a length exceeding all others is identified).

Inferences may also be detected at query processing time (see [9; 25; 12]). This approach may lead to long query processing since it most effectively operates on a user's entire query history (in conjunction with a logic-based inference engine).

Once the channels are identified we assign keys to users and tokens to the objects in the inference channels in such a way that users have flexibility in the objects they choose to access, provided the users are unable to complete the channel. Because the queries made by each user must affect the access capabilities of other users in order to ensure collusion resistance and the desirable property of crowd control, we draw inspiration for our schemes from secure group communication. In particular, the key allocation method employed in our schemes is similar to what's currently known as a *subset-cover* scheme [14] in the group communication setting. We believe that these cryptographic techniques are in many ways better suited to the inference control problem than the group communication setting. For example, if a large number of users have queried all but one object in an inference channel then it may be wise to consider this information to be in the public domain and to block all users (even those who have never queried the database) from querying the remaining object in the channel. The analogous situation in group communication is the revocation of an innocent user as the result of the revocation of several unauthorized users. This is a well-known problem in group communication that much work has gone into remedying (see, for example, [10]) yet it is a desirable property of an inference control scheme (what we call crowd control).

As mentioned in Section 1, an alternative approach to inference control that can be viewed as fitting in our general framework is to maintain a bit for each object indicating whether or not it has been released as the result of a query (see, for example, [9]). Our approach allows for more accurate access control because we can distinguish between n users each accessing $m-1$ objects in an inference channel of length m, and $n(m-1)$ users each accessing a single object in the channel. The method of [9] cannot, and this distinction is important since in the former case the remaining object in the channel should probably be blocked from all users if n is large, but in the latter case this may be unnecessary.

Finally, we note that if it is determined that a user's current query will enable them to make an undesired inference then this may be prevented by using query response modification (see, for example, [9; 23]). Our techniques can be used in conjunction with query response modification, that is, rather than requiring that the user receive higher authorization before completing the inference channel, it is also possible to simply modify the response, provided that still yields sufficiently useful information.

A survey of many of the existing approaches to inference control is in [4].

2. PRELIMINARIES

We denote the number of users of the database by n, and the users themselves by, $U_1, \ldots, U_n$. We view inference channels as being composed of objects. For our purposes, "object" is a generic term to describe a unit of information that is recoverable from the database, for example, a fact, attribute or relation. The example inference channel of Section 1 consists of objects that are each relations: the relation between ships and ports, and the relation between ports and weapons. We use m to denote the number of objects in an inference channel, and let $O_1, \ldots, O_m$ denote the objects in the channel. We sometimes refer to an inference channel of length m as an m-channel. The ship-port inference channel of Section 1 is a 2-channel.

In the initialization phase, users receive a set of encryption keys that they use to prove authorization (perhaps in addition to authenticating themselves to the database) to access the objects in an inference channel. Users prove this by encrypting some information specific to their object of interest. For example, the token might be an encryption of the attribute names needed to pose the query to the database. Referring back to the example of Section 1, we might require that the user form the token that is an encryption, under an approved key, of the attributes "ship name" and "port" concatenated together (i.e. the attributes are first represented as bit strings, then concatenated and then encrypted).[1]

We denote the set of keys allocated to U_i by K_i, $i = 1, \ldots, n$. Each encryption key may be known to several users. Before any queries have been made, each encryption key can potentially be used to access any object in the channel. However, users only have enough keys to generate tokens for a proper subset of the objects in the channel. We say a user has *max-*

[1]It is possible that with this approach a sequence of queries will be treated as though they form an inference channel when they do not. This is a common problem in inference control. We discuss ways to remedy this in Section 4.

imally queried the channel if they have used all possible keys to query the channel. By limiting the number of keys per user, we guarantee collusion resistance–a coalition of users cannot together query all the objects in the inference channel. The following definition makes this more precise.

Definition 1. *Let c be an integer, $0 < c \leq n$. We say that an inference protection scheme is c-collusion resistant if c users acting in collusion are unable to query all the objects in any inference channel.*

Once an encryption key is used to gain access to object O_i, the same key cannot be used to gain access to any object, O_j, $j \neq i$, in the same inference channel. This is accomplished by deleting (or, revoking) the tokens generated by that key from the rest of the inference channel. In other words, part of the automated access control mechanism is to update the list of acceptable tokens for objects in the inference channel each time an object in the channel is accessed (this processing time depends on the length of the channel). Hence, because a key may be used by many users, the queries of each user potentially affect the access capabilities of many users. For example, if two of the keys used to query an object in the channel belong to the same user, then this user will be unable to maximally query the channel. These properties allow us to achieve a property that we call *crowd control*: if a lot of users have queried a maximal portion of an inference channel (e.g. $m-1$ out of m objects) then no user should be able to complete the inference channel by making the remaining query. The reasoning here is that if an object has been queried a lot it is more likely to have been leaked and so it may be prudent to consider the query results to be in the public domain. So, if this is the case for most of the channel, access to the remaining object should be explicitly prohibited.

Definition 2. *Let $0 < \epsilon < 1$, and let U denote a randomly selected user. Consider a c-collusion resistant inference protection scheme. If when more than x sets of c users have together queried some set of $m-1$ objects, $O_{i_1}, \ldots, O_{i_{m-1}}$, in inference channel $\{O_1, \ldots, O_m\}$, then with probability at least $1-\epsilon$, U cannot access the remaining object, $\{O_1, \ldots, O_m\} - \{O_{i_1}, \ldots, O_{i_{m-1}}\}$, we say the scheme has (x, c, ϵ)-crowd control.*

Figure 1 is a high level example of how token sets, and consequently access control, changes as a result of user queries. The figure simplifies our approach in two ways. First, we don't show the keys each user receives, but rather just the tokens they generate from them (depicted here as ovals). The second, and more important, simplification is that tokens corresponding to different objects appear identical. In reality, the tokens are particular to the object with which they are associated, while the encryption keys used to generate them may be the same (see the discussion in Section 4 for more on this).

An important part of our analysis is assessing how information access changes as more users query the channel. To do this we need to understand how one user's set of keys relates to another user's set of keys. This is important because with each query the set of acceptable tokens for every other query can change. More precisely, we often study how much the key sets of one group of users, say $U_1, \ldots, U_r$, *cover* the key set of user, U_{r+1}. The size of the cover is the number of keys

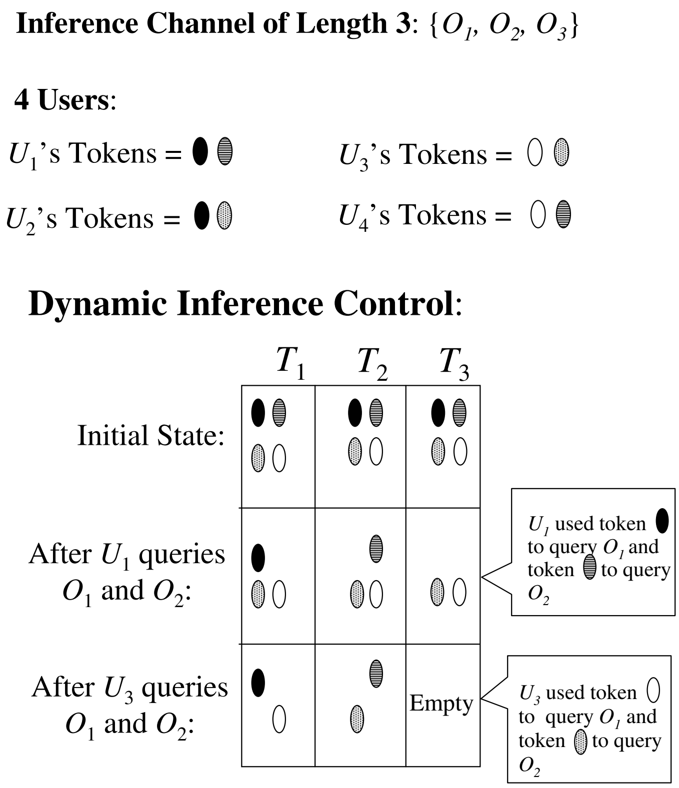

Figure 1: In this example there are four users each with two tokens (or, two keys that they use to generate tokens) and collusion resistance is $c = 1$. For $i = 1, 2, 3$, the ith column indicates which tokens can be used to access object O_i after the queries listed on the left hand side have been executed. After both U_1 and U_2 have queried objects O_1 and O_2, no one can query object O_3 (it has no more acceptable tokens) but everyone can still access both O_1 and O_2.

U_{r+1} has in common with at least one of $U_1, \ldots, U_r$, that is, the value: $|K_{r+1} \cap (\cup_{i=1}^{r} K_i)|$.

Finally, for simplicity of exposition, we often assume that a fractional quantity (i.e. $\frac{a}{b}$) is an integer. This should be clear from context.

3. DYNAMIC INFERENCE CONTROL

We assume that inference channels have been identified (for example, using a tool such as [17]) prior to the deployment of our inference control scheme. Our protocol consists of three phases:

- Key Allocation: Users are allocated $(m_{max}-1)/c$ keys, where m_{max} is the maximum length of an inference channel in the database, and c is the desired degree of collusion resistance.
- Initialization of the database: For each inference channel $\mathcal{Q} = \{O_1, \ldots, O_m\}$, a set of tokens, T_i, is associated with each object such that each user is capable of generating exactly $(m-1)/c$ tokens in T_i, for $i = 1, \ldots, m$. Initially, that is prior to any queries, the token sets are identical: $T_1 = T_2 = \ldots = T_m$.

- Query processing: If token $t \in T_i$ is used to gain access to O_i, then for every $s \neq i$, any token in T_s that was generated by the same key is deleted. Hence, the token sets change as queries are made.

We present two schemes that differ in how the first stage is completed; initialization of the database is essentially the same for both and query processing is as described above. The first is a simple randomized scheme that achieves probabilistic guarantees on crowd control (that is, $\epsilon > 0$). We analyze that scheme according to the crowd control and information access it permits as a function of the number of users querying the database.
We also present an algebraic scheme that offers deterministic guarantees on information access. Due to space constraints we do not analyze the scheme, although its analysis is similar to that of the randomized approach.

3.1 A Probabilistic Key Allocation Scheme

To allocate keys to the users we adopt a "bucket"-based approach that is often used in secure group communication (see, for example, [10]). Let there be $(m_{max} - 1)/c$ buckets, $B_1, \ldots, B_{(m_{max}-1)/c}$, each containing q encryption keys (that is, there are $q(m_{max} - 1)/c$ keys total). The keys themselves are randomly generated and are of a bit length suitable for the symmetric encryption scheme being used. For $i = 1, \ldots, n$, U_i receives a randomly selected key from each bucket for a total of $(m_{max} - 1)/c$ keys per user.
Token sets for an inference channel of length $m \leq m_{max}$ are formed by choosing a subset of $(m-1)/c$ buckets, $B_{i_1}, \ldots, B_{i_{\frac{m-1}{c}}}$, and using the keys in each bucket to generate the tokens for each object in the m-channel, as described in Section 2 (i.e. initially, every key in $B_{i_1} \cup \ldots \cup B_{i_{\frac{m-1}{c}}}$ can be used to access each object). Hence, before any queries are made, each user has the ability to query any $\frac{m-1}{c}$ objects, and so the scheme is c-collusion resistant. The following theorem shows how, for a given value of ϵ, the degree of crowd control afforded by the scheme depends on m and q. The idea behind the theorem is that a large set of users are likely to cover the key set of another user, so if these users have maximally queried the channel, the other user will be blocked.

Theorem 1. *Let $0 < \epsilon < 1$, and let c denote the collusion resistance of an instance of the probabilistic inference control scheme. Let $x = \frac{\ln(1-(1-\epsilon)^{c/(m-1)})}{\ln(1-c/q)}$ then the scheme has (x, c, ϵ)-crowd control.*

Proof. It suffices to show that if more than $x = \frac{\ln(1-(1-\epsilon)^{1/(m-1)})}{\ln(1-1/q)}$ sets of c users have queried $m-1$ objects, $O_{i_1}, \ldots, O_{i_{m-1}}$, in an inference channel of length m then the probability that a user U can query O_{i_m} is less than ϵ. U is unable to query O_{i_m} if the key sets of the users who have queried any of the other objects in the channel cover the relevant part of U's key set (i.e. those keys of U's that are useful for querying this channel). Of course, this happens with probability 1 if U has made $(m-1)/c$ other queries, and otherwise, the probability of this covering is at least $(1 - (1 - c/q)^x)^{(m-1)/c}$. Setting the latter quantity to be at least $1 - \epsilon$ and solving for x gives the quantity in the theorem statement. $\square$

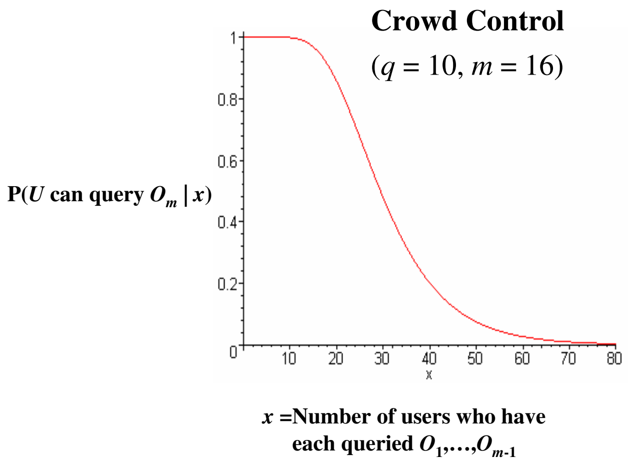

Figure 2: This figure shows how U's access to an object in an m-channel changes as the number of users who have each accessed the $m-1$ other objects in the channel grows. Here, $c = 1$, $q = 10$ and $m = 16$ (hence, this scheme can accommodate 10^{15} users). When the number of users is 70 or more, U can only access the object with probability at most .01.

Note that this theorem can be used to lower bound the probability that U can access object O_m when x sets of c users each have accessed *any* portion (i.e. not necessarily a maximal portion) of $O_1, \ldots, O_{m-1}$. However, it is a far weaker bound in this case. We claim that this is as it should be: if a large number of users have maximally queried the channel then for security purposes the remainder of the channel should be blocked, however if the users have only partially accessed the channel then the security risk to leaving it accessible is far less. To get a better idea of how access changes as a function of x, we include a concrete example of the access probabilities in Figure 2.
Theorem 1 shows that if a particular $(m-1)$-subset of the objects in a channel has been queried a lot, then users will be unable to query the *remaining* object in the channel whether or not they've already made some queries. It's likely that most users will still be able to access some $((m-1)/c)$-subset of the queried objects, however, as the following lemma shows.

Lemma 1. *Let $0 < \epsilon < 1$, and let c denote the collusion resistance of an instance of the probabilistic inference control scheme. If $x > \frac{\ln(1-(1-\epsilon)^{c^2/m^2})}{\ln(1-1/q^2)}$ users have each maximally queried an m-channel, then with probability greater than $1 - \epsilon$, a user U, who is not amongst the x users, can maximally query the same channel.*

Proof. A user U cannot maximally query the channel if two of U's keys have been used to query a single object. This is impossible if for every pair of U's keys, one of the users who has maximally queried the channel has both such keys. The probability of this is at *least* (note this is a coarse bound): $(1 - (1 - 1/q^2)^x)^{(m/c)^2}$. Setting this quantity to be greater than $1 - \epsilon$, we get the bound in the lemma statement. $\square$

The above results demonstrate how a threshold number of queries to a channel affect the access capabilities of other users, but they don't describe how information access changes before the threshold is reached. We prove a lower bound on information access as a function of the number of users querying the channel by using the fact that any of U's keys that have *not* been used to query an object in the channel, may be used to query *any* object in the channel.

THEOREM 2. *Let $0 < \alpha < 1$. Consider a c-collusion resistant instance of the probabilistic inference control scheme. If the number of users who have maximally queried the channel is $x < \frac{q(1-\alpha)\epsilon^{c/((m-1)(1-\alpha))}}{e}$ then with probability at least $1-\epsilon$, another user can access any subset of objects in the channel of size $\frac{\alpha(m-1)}{c}$.*

PROOF. Consider a set of $x+1$ users, $U \notin \{U_1, \ldots, U_x\}$, the expected number of U's keys that $U_1, \ldots, U_x$ cover, is $\frac{(m-1)x}{cq}$. We show that the probability that $U_1, \ldots, U_x$ cover more than $1-\alpha$ of the keys U has for querying the channel is less than ϵ. From Chernoff bounds [13], it follows that this is true when the following inequality holds: $\left(\frac{e^{(q(1-\alpha)/x)-1}}{(\frac{q(1-\alpha)}{x})^{\frac{q(1-\alpha)}{x}}}\right)^{\frac{(m-1)x}{cq}} < \epsilon$. This inequality is satisfied by the x bound given in the theorem statement. □

To reduce user U's access to an inference channel, two users sharing distinct keys with U must use these keys to query the same object, O, in the channel. In such a situation, U is unable to use either key to query other objects in the channel, while U only needs one of the keys to query O, hence one of U's keys cannot be used. Hence, restricting a user's access to the channel requires more about the key allocation structure than we use in Theorem 2, and so we suspect that this lower bound is generally not tight (and Lemma 1 provides particular evidence that it's not tight).

3.2 A Variant with Deterministic Guarantees

The key allocation scheme of Section 3.1 guarantees crowd control and information access probabilistically as a function of the number of users querying an inference channel. In some settings deterministic guarantees are necessary. We can achieve some deterministic guarantees by using error-correcting codes to allocate keys to users. Specifically, we use Reed-Solomon codes (see, for example, [27]) to allocate keys to users. Reed-Solomon codes use a polynomial that's unique to each user to determine each user's codeword, or in our case, key set. Because two polynomials of the same degree intersect on at most as many points as their degree, two users in our inference control scheme will share at most as many keys as the degree of their polynomials. Using this fact, we construct a scheme with deterministic (i.e. $\epsilon = 0$) information access guarantees. The following makes the key allocation part of such a scheme more precise, the initialization of the database and the query processing are both just as before.

We consider all polynomials of degree t, $0 < t < (m_{min} - 1)/c$, over the finite field F_q of q elements, where m_{min} is the minimum length of an inference channel. To each user, U, we associate a unique such polynomial, $p_U(x) \in F_q[x]$. For each element $(\gamma, \beta) \in F_q \times F_q$ we generate a random key, $k_{\gamma,\beta}$ of the desired bit length. User U receives the set of keys $K_U = \{k_{\gamma,p_U(\gamma)} | \gamma \in A \subseteq F_q\}$, where A is a set of size $(m_{max} - 1)/c$. Note that this is very similar to the bucket-based construction of Section 3.1 except that using polynomials to allocate keys from the "buckets" gives more control over the overlap between users' key sets. The following lemma demonstrates one of the deterministic guarantees provided.

LEMMA 2. *Consider a c-collusion resistant inference protection scheme that uses the key allocation method of this section. If $x < \frac{(m-1)(1-\alpha)}{ct}$ users have maximally accessed an m-channel then another user can access any subset of objects in the channel of size $\frac{\alpha(m-1)}{c}$ with probability 1.*

PROOF. Consider a user U who is not among the x users who have maximally accessed the channel. U shares at most t keys with each of the x users, and so at most $tx < t(\frac{(m-1)(1-\alpha)}{ct}) = \frac{(m-1)(1-\alpha)}{c}$ keys total. Hence, U has more than $\frac{\alpha(m-1)}{c}$ keys that none of the x users have and can access a different object in the channel with each key. □

An analysis very similar (but a bit more involved due to the fact that keys aren't assigned independently) can be performed to prove crowd control and lower bounds on information access. The scheme performs comparably to the earlier one.

4. DISCUSSION AND OPEN PROBLEMS

We have concentrated on a single inference channel for simplicity of exposition. When using our methods to prevent inferences across multiple channels a potential problem arises when a single object appears in channels of different lengths. To ensure that no inferences can be made by exploiting the varying channel lengths it may be necessary to reduce the number of acceptable keys associated with objects in overlapping channels, thus reducing information access.

In addition to focusing on a single channel we have also assumed a fixed user base. Over time, however, it is likely that users will lose access to the database (i.e. be revoked) and new users will be added. The scheme of Section 3.1 has a fixed upper bound on the number of users it can accommodate: $q^{(m_{max}-1)/c}$ (the bound for the scheme of Section 3.2 may be less depending on t). To allow for the addition of new users we can choose q to be larger than is currently required. Of course, this will have consequences for crowd control: increasing q means users' key sets are more disjoint and so the queries of individual users tend to have less of an impact on the access capabilities of others.

When a user is revoked from accessing the database their keys must no longer be valid. Simply deleting the tokens generated from their keys might unfairly restrict the access of others, so rekeying of the valid users may be needed. There is a large body of work on revocation in group communication that we may leverage for this problem. For example, efficient techniques for key updating over a broadcast channel are provided in [16; 10]. In addition, we note that protection against key leaks can be built into the key allocation scheme by ensuring that if a group of users pool their keys a leak a subset of the resulting pooled set, we will be able to *trace* at least one of the culprit users. Such tracing capability can be incorporated into exactly the type of key allocation we use in this paper (see, for example, [19]) but it will tend to increase the crowd control threshold, x.

As mentioned earlier (and as Lemma 1 indicates) our lower bound on information access for the scheme given in Theorem 2 isn't tight. A proof that takes more of the combinatorics of the key allocation scheme into account should produce a better bound. In addition, there may be ways to reduce user storage while retaining inference control. Using similar techniques to those in [14], it may be possible to allocate to each user a smaller set of keys that they can then use to generate additional keys, and thus, tokens.

Our approach offers more flexible information access than existing approaches with fast query processing, but it still may not offer quite the flexibility afforded by schemes that rely on user query histories to determine access control. This is because, as mentioned in Section 1.1, we require users to form tokens that are derived from information that's specific, but not necessarily unique, to the object of the user's interest. We could easily base the tokens on unique information but we think that doing so will increase the total number of inference channels so much as to negatively impact performance.

Finally, we note that with our schemes it is possible to determine what information the user *could* have accessed just from the current access control state (i.e. the value of the token sets at various points in time). For example, any key that appears in a token set T_i and no others must have been used to query object O_i, whereas a key that appears in all the token sets could not have been used. Hence, if a user is known to be compromised, the authorities can use their knowledge of the user's keys to determine what information the user *could* have accessed. Of course, provided the users' keys sets are kept private unless such a security breach occurs, the fact that keys are shared amongst users also provides desirable privacy to the users of the database.

Acknowledgements

The author is grateful to Teresa Lunt for numerous helpful discussions and to David Woodruff for comments on an earlier draft of this paper.

5. REFERENCES

[1] A. Barani-Dastjerdi, J. Pieprzyk and R. Safavi-Naini. *Security in databases: A survey study.* Manuscript, 1996.

[2] L. Binns. *Implementation considerations for inference detection: Intended vs. actual classification.* In Proceedings of the IFIP WG 11.3 Seventh Annual Working Conference on Database Security, 1993.

[3] D. Denning and T. Lunt. *A multilevel relational data model.* In IEEE Symposium on Security and Privacy, 1987.

[4] S. Jajodia and C. Meadows. *Inference problems in multilevel secure database management systems.* In Information Security: An Integrated Collection of Essays, M. Abrams et al., eds., IEEE Computer Society Press (1995), pages 570-584.

[5] J. Hale and S. Shenoi. *Catalytic inference analysis: Detecting inference threats due to knowledge discovery.* In the IEEE Symposium on Security and Privacy, 1997, pp. 188-199.

[6] T. Hinke. *Database inference engine design approach.* In Database Security II: Status and Prospects, 1990.

[7] T. Hinke, H. Degulach and A. Chandrasekhar. *A fast algorithm for detecting second paths in database inference analysis.* Journal of Computer Security, 1995.

[8] T. Hinke, H. Degulach and A. Chandrasekhar. *Layered knowledge chunks for database inference.* In Database Security VII: Status and Prospects, 1994.

[9] T. Keefe, M. Thuraisingham, W. Tsai. *Secure query-processing strategies.* IEEE Computer, Vol. 22, No. 3, 1989, pp. 63-70.

[10] R. Kumar, S. Rajagopalan and A. Sahai. *Coding constructions for blacklisting problems without computational constructions.* In Advances in Cryptology – Crypto '99, pp. 609-623.

[11] T. Lunt. *Access control policies for database systems.* In Database Security II: Status and Prospects, pp.41-52.

[12] D. Marks, A. Motro, S. Jajodia. *Enhancing the controlled disclosure of sensitive information.* In Proc. European Symp. on Research in Computer Security, Rome, Italy, September, 1996.

[13] R. Motwani and P. Raghavan. *Randomized Algorithms.* Cambridge University Press, 2000.

[14] D. Naor, M. Naor and J. Lotspiech. *Revocation and tracing schemes for stateless receivers.* In Advances in Cryptology – Crypto 2001.

[15] T. Lin. *Inference secure multilevel databases.* In Database Security VI, pp.317-333.

[16] M. Naor and B. Pinkas. *Efficient trace and revoke schemes.* In Financial Cryptography 2000.

[17] X. Qian, M. Stickel, P. Karp, T. Lunt and T. Garvey. *Detection and elimination of inference channels in multilevel relational database systems.* In IEEE Symposium on Security and Privacy, 1993.

[18] G. Smith. *Modeling security-relevant data semantics.* In IEEE Symposium on Security and Privacy, 1990.

[19] D. Stinson and R. Wei. *Key preassigned traceability schemes for broadcast encryption.* Lecture Notes in Computer Science 1556 (1999), 144-156 (SAC '98 Proceedings).

[20] B. Sowerbutts and S. Cordingley. *Database architectonics and inferential security.* In Database Security IV, pp. 309-324.

[21] M. Stickel. *Elimination of inference channels by optimal upgrading.* In IEEE Symposium on Security and Privacy, 1994.

[22] B. Thuraisingham. *Data mining, national security, privacy and civil liberties.* To appear in Knowledge Discovery and Data Mining Magazine (SIG KDD), 2003.

[23] B. Thuraisingham. *Mandatory security in object-oriented database systems.* In proceedings of OOPSLA '89.

[24] B. Thuraisingham. *Recursion theoretic properties of the inference problem.* In the IEEE Third Computer Security Foundations Workshop, 1990.

[25] B. Thuraisingham. *Towards the design of a secure data/knowledge base management system.* In Data Knowledge and Engineering, 1990.

[26] B. Thuraisingham. *The use of conceptual structures for handling the inference problem.* In Database Security V, pp.333-362.

[27] J. van Lint. *Introduction to Coding Theory.* Springer-Verlag, 1999.

[28] R. Yip and K. Levitt. *Data level inference detection in database systems.* In the IEEE Eleventh Computer Security Foundations Workshop, 1998.