

Mohammed Al-Zoubi  
Supervisors:  
Dr. Seyed Shahrestani and Dr. Chun Ruan

**Sensitivity Level Base Model with RBAC access Control (SLB-RBAC)**

Contents

[1 Summary 4](#_Toc434893613)

[2 Introduction 5](#_Toc434893614)

[3 Quasi-Identifiers (QID) and k-Anonymity 7](#_Toc434893615)

[4 Choosing Quasi Identifier 7](#_Toc434893616)

[5 Differential Privacy 9](#_Toc434893617)

[6 (X,Y) Anonymity 10](#_Toc434893618)

[7 ℓ-Diversity 10](#_Toc434893619)

[8 High Dimensional Data 11](#_Toc434893620)

[9 LKC-Privacy 12](#_Toc434893621)

[10 Generalization 13](#_Toc434893622)

[11 Top-Down Specialization TDS 16](#_Toc434893623)

[12 MapReduce 19](#_Toc434893624)

[Airavat 19](#_Toc434893625)

[GUPT 20](#_Toc434893626)

[13 Problems with the current Analytics Models 21](#_Toc434893627)

[14 My Contribution 22](#_Toc434893628)

[Preliminary 22](#_Toc434893629)

[SLB-RBAC Method 22](#_Toc434893630)

[Definitions 22](#_Toc434893631)

[15 Equations 23](#_Toc434893632)

[The Sensitivity Level Equation 23](#_Toc434893633)

[The Generalization Level Equation 23](#_Toc434893634)

[24](#_Toc434893635)

[24](#_Toc434893636)

[Ownership Level (*ol*) 24](#_Toc434893637)

[The Object Aging Sensitivity (Ṑ) 26](#_Toc434893638)

[Calculating the SLB Example 26](#_Toc434893639)

[SLB specifications 29](#_Toc434893640)

[16 RBAC Mapping 30](#_Toc434893641)

[17 SLB-RBAC Framework 32](#_Toc434893642)

[Security Assertion Markup Language (SAML) 32](#_Toc434893643)

[SAML and XML 33](#_Toc434893644)

[SLB and SAML 36](#_Toc434893645)

[18 SLB Experiment 39](#_Toc434893646)

[19 Conclusion 44](#_Toc434893647)

[20 References 45](#_Toc434893648)

# Summary

*Concurrent to the swift developments in Big Data and its related services, an emerging trend has been developed to use the data analytics in large-scale environments. This has led to finding an access control model with a scale-up ability to cope with the big data growth. Different models have been proposed with a limited Mandatory Access Control (MAC) that is unable to scale-up parallel with big data, and hard to integrate with the access control model owned by data owners. Hence, MAC contaminates the principle of authorization over the cloud, since the cloud access is mandated by a large number of user’s accesses.*

*A number of papers and k-anonymity models have been reviewed in this report, started from the analytics models, moving toward outlining the base of the novel Sensitivity Level Base Model (SLB). This report focuses on using RBAC access control to control and secure access for data analytics in big data. General review for generalization models and k-anonymity is introduced, then a proposed contribution model. The SLB-RBAC model needs to be tested and amended accordingly. Once the algorithms and the equations are verified and compared with similar models, the next step will move forward toward scaling up the model for a real data environment.*

# Introduction

Big Data implies special care in the matter of scaling up data storing and retrieval on need. Data analytics is the major anatomy in big data concern, MapReduce may be exploited by developers in breaching data privacy. Many security algorithms have been developed in the recent few years to tackle the data leakage concern in MapReduce.

Before focusing on MapReduce major, it is worth discussing the possible attacks in data analytics process. It is possible to hide some sensitive information from the attackers, for instance person’s name, address, contact number and email, but in some cases, it is hard to stop data leakage by gaining some data indirectly, using results output and comparing it with other external data.

Side attack in data analytics can be divided into three types of attacks; state attack, privacy attack, and timing attack. The state attack can be triggered by the adversary code, the code changes the values of static variable, such as the keyword. In this case the privacy algorithms may lose the protection control. The attacker may run malicious code to transfer the other mapper’s output through the network. Another popular attack is the privacy, when the adversary reads some data and compares it with his/her external data. It is not necessary for the adversary to reach the sensitive data, it can be predicted based on the other attributes. Finally, the timing attack is possible by using an infinitive loop in the script, or by forcing scripts to run longer than the expected time.

The user keyword is also a prone for attacks. The search keyword can be derived by using different techniques such as; output values, the key it outputs, the order of output keys, or relationship between the output keys and the values.

Side information may occur when the malicious user reads some queries on other attributes so he/she can determine if the required person has a diabetes or not, for instance. The below table shows an example of a dataset in a table, the MapReducer is permitted to read all attributes except the patient name. Since the user knows some information about Steven such as his age, then it is obvious that finding the summation of people who have a diabetes is a straight forward, this can be done by querying two queries, as in the query of Q(6) – Q(5), let Q(6)=“find all users who have diabetes except Karen”, and the second query Q(5)= “find all users who have diabetes”, with comparing these two queries, possible attack may occur against Karen’s private information [1].

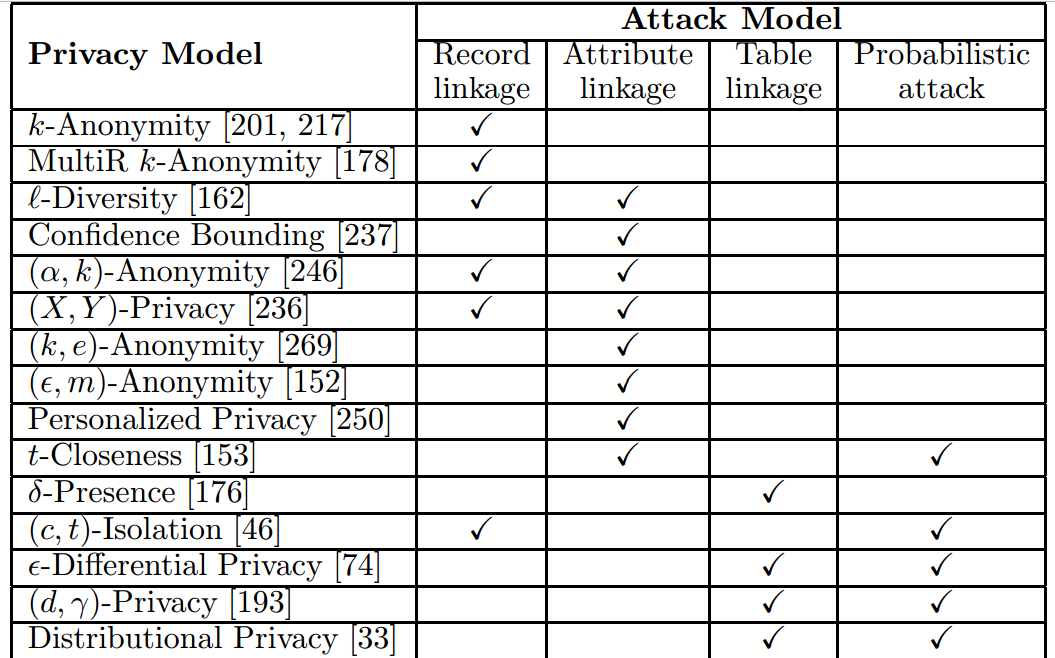
Table 1 Security attack using side information

|  |  |  |  |
| --- | --- | --- | --- |
| **Patient\_Age** | **Gender** | **Name** | **Has Diabetes** |
| 45 | Female | Marry | 1 |
| 40 | Male | Paul | 1 |
| 38 | Male | Mark | 0 |
| 55 | Female | Karen | 1 |
| 62 | Female | Nicole | 0 |
| 41 | Male | Steven | 1 |

Private Data leakage occurs in different scenarios, the above side reading is one of them. Another example is the changing of requested attributes and the combination is another worry, the query may request “Patient Age” together with “Gender”, which is more possible to expose information about patients.

Different privacy models are proposed by researchers such as; k-anonymity, l-diversity, and confidence bounding. The privacy models attempt to stop data leakage in the data object level, which includes; tables, records (tuple), attributes, and files. Table 2 shows verities of privacy models and the possible attack model prepared by [2] . The comparison shows the attribute linkage to external tables is the highest.

Table 2 shows different privacy models with the attack possibility



# Quasi-Identifiers (QID) and k-Anonymity

One of the privacy techniques is the Quasi-Identifier, it implies finding a group of attributes that can identify other tuples in the database. These identifiers may not gain 100% of data, but even though, a risk of predicting some data remains high. In the above example, knowing the patient age, gender, and postcode, may lead to uniquely identifying that patient with 87%.

Many algorithms have been developed to overcome this security breach. K-anonymity was developed by Sweeney [3], who proposed generalization and suppression for quasi-identifier (QID) attack. *K-anonymity* guarantees a privacy on releasing any record by adhering each record to at least k individuals, this is correct even if the released records are connected to external information. The table is called k-anonymous, if one tuple has QID values, and at least k – 1, other records also have QID values. This means, the minimum equivalence group size on QID is at least k.

Sweeney method starts from the QID *K-anonymity*, which defines any table RT(*A1,…,An*), associated with the QID, is said to be *K-anonymity* if each sequence of values in RT appears k times.

Sweeney identifies both generalization and anonymization solutions for Quasi-Identifier. The principle of the definition adheres each QID is defined by a group domains, this implies attributes {A1,…Aj} in the table TR, each value in the table appears with a sequence of K occurrence.

Example: the following table show patient’s details, the next following definitions explain Sweeney solution:

Table 3 Sweeney definitions QID={Patient\_age,Race,Postcode}, and K=3

|  |  |  |  |
| --- | --- | --- | --- |
| **Patient\_Age** | **Race** | **Postcode** | **Diabetes\_Type** |
| 56 | Anglo | 2000 | 2 |
| 50 | South American | 2101 | 1 |
| 45 | African | 2000 | 1 |
| 45 | Anglo | 2015 | 2 |
| 56 | Anglo | 2000 | 2 |
| 50 | South American | 2015 | 1 |
| 56 | African | 2101 | 0 |
| 50 | South American | 2015 | 1 |
| 45 | African | 2101 | 2 |

The above example shows Sweeney definition comprises IQ={Patient,Race,Postcode} with k=3, means any two of the four QID attributes may incur a possible attack. Each value in IQ appears at least 3 times, Age={56,45,50}, Race={Anglo,South American,African}, and Postcode={2000,2101,2015}. This example satisfies *k-anonymity* with respect to QID.

# Choosing Quasi Identifier

Data set is used in many privacy models, all models need to identify either Quasi Identifier QID, sensitive-Attributes, or non-Sensitive Attributes. The QID should be identified first, and before categorizing attributes to sensitive and non-sensitive. In reality, there is no definite answer for such a step, especially that the definition of QID and sensitivity depends on external sources of data. One of the proposed algorithms was released on 2007 by Motwani and Xu [4], the algorithm used the terms of separation minimum key ratio (ε) and distinct minimum key ration (δ). Researchers try to find a QID that protects the privacy and minimizes the data loss during the anonymization. The proposed algorithm endeavour to find a small QID with provable size and separation, named as “Greedy” algorithm. This algorithm is useful in the large tables which consists of millions of records.

The minimal key is used in this algorithm by stating both of distinct and separation. The target is finding the minimum key for QID, this can be found using the greedy minimum key algorithm, which outputs a key of size (1 + 2ln n), where n is number of records. This algorithm scans the table twice, which is not practical in the large data size, therefore, a lighter algorithm was developed. The algorithm randomly samples a set of tuples, and reduces the input set cover (key) to a smaller set, containing only the chosen tuples.

The algorithms are able to find QID, based on greedy algorithm. However, this algorithm is expensive and slow, otherwise naïve greedy algorithms are unable to accurately find the QID. Moreover, the Greedy algorithm has ignored the external attributes that adversary may have, so he/she can connect between them.

# Differential Privacy

Many methods were suggested to anonymize the data on running analytics queries, data owner wish to hide private information from the data analysts. This approach creates issues in data accuracy. One of these methods termed as Differential Privacy, this method depends on the analyser or (curator) who use whether interactive or non-interactive data analytics, the curator resides in the middle between the data and the user, the curator receives the user query in the interactive type, amend it for security and privacy, so the data can’t be destroyed as it is real data.

Many algorithms have been constructed by researchers, however, all algorithms use the principle of adding noise to data output. The noise should not produce considerable changes to the actual data. For a clear understanding, let us consider the following example:

Table 4. Database table D1 Table 5. Database table D2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Has Diabetes** |  | **Name** | **Has Diabetes** |
| Marry | 1 |  | Marry | 1 |
| Paul | 1 |  | Paul | 1 |
| Mark | 0 |  | Mark | 0 |
| **Karen** | **1** |  | **Karen** | **0** |
| Nicole | 0 |  | Nicole | 0 |
| Steven | 1 |  | Steven | 1 |

Suppose that a database D1, shown in table 4, and database D2, as shown in table 5. If a curator needs to know Karen’s diabetes status, then it is possible by running two queries, as previously described. If the output value was changed for one value in the table, named as singleton, then the database table D2 will be created after the change has occurred. Note that Karen=1, and Karen=0 makes a different in Karen’s status, since the curator shouldn’t know individual data, and the collected and analysed data must be used statistics rather than patient’s private information.

Suppose that a query needs to calculate the number of patients with diabetes, the query result is (4) for D1, and (3) for D2. The output result can be anonymised using some algorithms to create a perturbation and alters the output to (Q(D1)=3.8), and (Q(D2)=3.6). These two values are very close, and setting Karen to zero, or omitting her will not make any difference. An exponential value can be used to create this perturbation as shown in this definition [5]:

Definition

*Pr[K(D1) ∈ S] ≤ exp(£) × Pr[K(D2) ∈ S]*

Where K: is randomized function, *£*: deferential privacy. S ⊆ Range(K), Pr: Probability of tossing coins

The above definition is valid for a single value. If more than one record need to be anonymised then different definitions are used. The definition is proposed as the Laplace Distribution. The Laplace Distribution (centered at 0) with scale b is the distribution with probability density function:

Lap(x|b) = exp ( − || )

The variance of this distribution is σ 2 = 2b 2 [1]

# (X,Y) Anonymity

Wang and Fung proposed (X,Y) Anonymity, were X and Y are disjoint set of attributes, where x represent the attributes, and Y represent the sensitive attribute. This model is useful when several records may represent the same owner. Therefore, the k-anonymity is a special case of (X,Y) anonymity. It is most likely to have the table primary key as a value of Y, so the sensitivity is distinguished by the primary key.

# ℓ-Diversity

The *ℓ-diversity* is introduced by Machanavajjhala et al [6]. This algorithm aims to reduce the attributes linkage. It is developed from the fact that some sensitive attributes [S] are more frequent than others in the group. So the ℓ-diverse is calculated using the entropy, by grouping the QID and then calculating the entropy for the groups. Using the following:

For example let us consider table 6, as a part of age generalisation, and IQ={Patient\_Age,Race,Gender}, while the sensitive attribute is Disease. The records are grouped or compressed with the similar age, race, gender, and disease. Based on the above given definition, the entropy can be calculated as:

Table 6, illustrates *ℓ-diversity* model

|  |  |  |  |
| --- | --- | --- | --- |
| **Patient\_Age** | **Race** | **Gender** | **Disease** |
| 25-30 | Anglo | Male | Flu |
| 25-30 | Anglo | Male | Flu |
| 25-30 | Anglo | Male | Eczema |
| 30-35 | South American | Female | [Thalassemia](https://www.google.com.au/search?safe=off&biw=1242&bih=599&q=thalassemia&spell=1&sa=X&sqi=2&ved=0CBkQvwUoAGoVChMIsKn34oj-xwIVRuemCh3xdAXB) |
| 30-35 | South American | Female | [Thalassemia](https://www.google.com.au/search?safe=off&biw=1242&bih=599&q=thalassemia&spell=1&sa=X&sqi=2&ved=0CBkQvwUoAGoVChMIsKn34oj-xwIVRuemCh3xdAXB) |
| 30-35 | South American | Female | [Thalassemia](https://www.google.com.au/search?safe=off&biw=1242&bih=599&q=thalassemia&spell=1&sa=X&sqi=2&ved=0CBkQvwUoAGoVChMIsKn34oj-xwIVRuemCh3xdAXB) |
| 30-35 | South American | Female | Pneumonia |

The group <[25-30],Anglo,Male>

The group <[30-35],South American,Female>

To achieve entropy ℓ-diversity, the table as a whole must be at least log(ℓ) since the entropy of a QID group is always greater than or equal to the minimum value.

The minimum value of entropy ℓ -diversity =1.8, is considered to be the lowest value for the whole table.

Eventually, the entropy is used in to anonymize the sender in communication system. This technique is used to defend the adversary who applies the traffic analysis to identify the sender ID. Mix networks and crowd are used to hide the sender ID using entropy.

The entropy ℓ -diversity is not effective in the real data environment, since grouping the similar records does not reduce the adversary attack possibility. In the above example, grouping the South American who have Thalassemia does not convey the possible successful percentage of 75%. The ℓ -diversity doesn’t provide any measurement for portability-based risk.

One of the Big Data features is containing a large size of objects, users queries are infinitive and difficult to predict. Queries with a large size of objects may contain a high dimensional data. For example social networks, financial and health organizations use analytics with sophisticated queries. ℓ -diversity and k-anonymity suffer from the curse of dimensionality.

# High Dimensional Data

In classical database, objects are used to store data, these objects are tables, indexes, sequences, xml, files, views, and synonyms. Objects contain number of records or tuples, and each record presents the same set of attributes. Each key can be distinguished by a primary key, and a secondary key connects the objects with other objects, secondary or primary key is usually numerical. Each piece of data is named a cell, the cell value may contains numerical of both integers and real (integers are discrete and real are continuous), alphanumerical, text, binary, or characters. Some data are discrete, while others are continuous. Objects may remain static, while other shrink or expand dynamically. Some data are extremely large and hard to fit the computation memory, hence provisioning is needed.

The above description show the verities of data types that data analytics deals with. One of the high cost computation data is the multi-dimensional data. Many fields use the high dimensional data such as patterns recognition, images, and general medical data. Data analytics encounters search methods to find a group of records, these records could be a part of one-dimensional data, two-dimensional data, or multidimensional data [7].

Proper algorithms should be used to provide accurate queries results. Consider many objects about human genes are available for analytics, suppose that we need to create a classifier to read any person’s genetic code, and the possible future diseases. The analyser matches between the person genes and the other patients who carry close similar DNA. The analyser first takes one dimention feature like the blue eye color gene, to find out the expected diseases for people with blue eyes, this type of search is one dimensional. This description of eyes is not enough, therefore more descriptions can be added to give more accurate results. More added DNA description, means better results. However, this is not always true, as shown in figure 1, the relationship between descriptions and dimensionality, increases to a certain level then it declines exponentially. Hence, increasing the number of dimensions may cause degraded values, this degradation is named “curse of dimensionality” as a result of overfitting.

More objects added are also useless, and

Multidimensional data tends to find neighbours objects that have particular feature values, the feature values depends on the required query. Some queries are more complicated than other, and as a result more computation time. The search aims to find similar objects or to match corresponding objects in two or more sets of objects so that the result is a pairing of all of the objects from one set with objects in the other set. Solving queries depends in multidimensional data depends the used algorithms, there are complicated queries that cannot be solved by the provided algorithms. Therefore, Bellman has coined this problem and named as “the curse of dimensionality”.

# LKC-Privacy

The model can be applied for the multidimensional data, such as patient’s information. General intuition of LKC-privacy insures that QID with a length of L and sensitive value of S is not greater than Class C, the idea is grouping length of records L in the data object T, by at least k records.

The following example illustrates the LKC-Privacy. Suppose the following table 7, and taxonomy figure 1, where L=2, K=2, and C=50% (Yes or No). The table 6 was generalized using Figure 1 taxonomy. Based on the given information, let us determine whether the generalization in table 6 is correct or not, in relate to LKC-Privacy model.

Table 7, illustrates LKC-Privacy model

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | *Quasi-identifier (QID)* | | | *Class* | *Sensitive* |
| **ID** | **Job** | **Sex** | **Age** | **Transfuse** | **Surgery** |
| 1 | Cleaner | F | 35 | Y | Appendicitis |
| 2 | Cashier | F | 31 | Y | Appendicitis |
| 3 | Teacher | M | 35 | N | Urology |
| 4 | Engineer | M | 27 | N | Urology |
| 5 | Plumber | M | 25 | Y | Vascular |
| 6 | Electrician | M | 29 | N | Vascular |

Table 8, the previous table has been generalized for (Job, Age)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | *Quasi-identifier (QID)* | | | *Class* | *Sensitive* |
| **ID** | **Job** | **Sex** | **Age** | **Transfuse** | **Surgery** |
| 1 | Non-Technical | F | 30-60 | Y | Appendicitis |
| 2 | Non-Technical | F | 30-60 | Y | Appendicitis |
| 3 | Professional | M | 30-60 | N | Urology |
| 4 | Professional | M | 1-30 | N | Urology |
| 5 | Technical | M | 1-30 | Y | Vascular |
| 6 | Technical | M | 1-30 | N | Vascular |

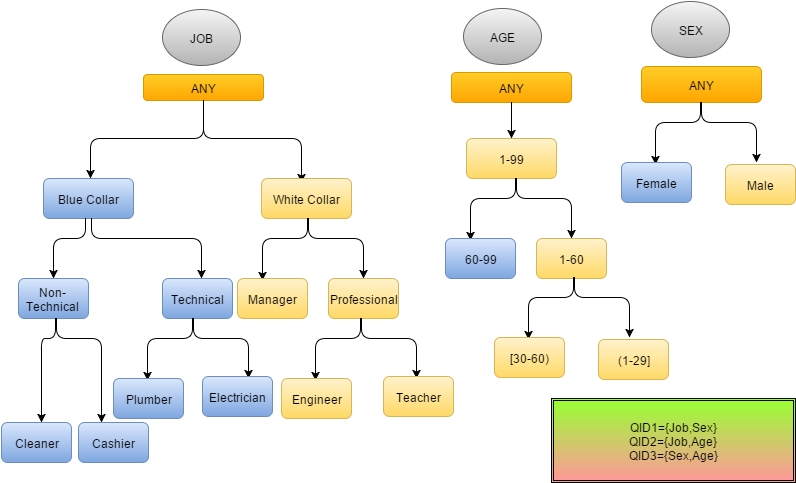


Figure 1 illustrates Taxonomy Trees

As shown in table 8, two records can be grouped together, so the generalization occurred in QID2 that supports the records grouping. Only one record for <professional,M,30-60> can’t be grouped with the similar record, as the age interval is different. This implies another generalization level on age, for example between 0-90, which results higher utility loss.

The cut generalization for the above example is not successful, therefore, it is better find different method generalization.

# Generalization

The general method defines Minimum Generalization (MinGen), and Maximum Generalization (MaxGen).in the above example, if the curator request a query with two QID attributes, then the MinGen can be represented by omitting some values, or replacing them. The MaxGen implies values suppression, or completely hiding the values.

The following relationship implies the existence of the Value Generalization Hierarchy VGH for any attribute A for the function (*f*).

The generalization is defined as:

The relationship defines a Domain Generalization Hierarchy DGH for an Attribute A, in a tuple t(An), comparing to Attributes (A1,…An). The generalization g for the table T is defined as g(T). The generalization level (z), depends on the attribute value (*νi*), some values can be generalized up to three level before suppression is occurred like postcode in the following example:

Table 9, illustrates PT as a part of RT

|  |  |
| --- | --- |
| **Race** | **Postcode** |
| Anglo | 2100 |
| South American | 2109 |
| African | 2100 |
| Anglo | 2175 |
| Anglo | 2109 |
| South American | 2175 |
| African | 2100 |
| South American | 2175 |
| African | 2109 |

Postcode generalization DGH(Postcode): Z0(2100,2109,2175), Z1(210\*,217\*), Z2(21\*\*), Z3(\*\*\*\*).

Race Generalization DGH(Race): Z0(Anglo,South American,African), Z1(person), Z2(\*\*\*\*\*\*).

The generalized tables results are: GT(1,0), GT(1,1), GT(0,2), GT(0,1), as show in tables 10. Notice that the GT(3,2), GT(2,2) and others are not possible in generalization, as they are assigned on suppression.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Race (E0)** | **P.Code**  **(Z0)** |  | **Race (E1)** | **P.Code (Z0)** |  | **Race (E1)** | **P.Cod(Z1)** |  | **Race (E0)** | **Postcode**  **(Z2)** |  | **Race**  **(E0)** | **Postcode**  **(Z1)** |
| Anglo | 2100 |  | Person | 2100 |  | Person | 210\* |  | Anglo | 21\*\* |  | Anglo | 210\* |
| S.Amer | 2109 |  | Person | 2109 |  | Person | 210\* |  | S. Amer. | 21\*\* |  | S. Amer. | 210\* |
| African | 2100 |  | Person | 2100 |  | Person | 210\* |  | African | 21\*\* |  | African | 210\* |
| Anglo | 2175 |  | Person | 2175 |  | Person | 217\* |  | Anglo | 21\*\* |  | Anglo | 217\* |
| Anglo | 2109 |  | Person | 2109 |  | Person | 210\* |  | Anglo | 21\*\* |  | Anglo | 210\* |
| S.Amer | 2175 |  | Person | 2175 |  | Person | 217\* |  | S. Amer. | 21\*\* |  | S. Amer. | 217\* |
| African | 2100 |  | Person | 2100 |  | Person | 210\* |  | African | 21\*\* |  | African | 210\* |
| S.Amer | 2175 |  | Person | 2175 |  | Person | 217\* |  | S. Amer. | 21\*\* |  | S. Amer. | 217\* |
| African | 2109 |  | Person | 2109 |  | Person | 210\* |  | African | 21\*\* |  | African | 210\* |

PT GT(1,0) GT(1,1) GT(0,2) GT(0,1)

Tables 10 Generalized tables GT

The above tables can be distinguished by the precision value, the higher precision is the chosen table.

The precision table can be calculated using the following equation:

Where DGH(Postcode)=3, and DGH(Race)=2, also PT=2, and NA=9

The above tables precision results are:

Prec(GT(1,0)=1- 9/2/18 = 0.75

Prec(GT(1,1)=1- (9/3 + 9/2)/18 = 0.58

Prec(GT(0,1)=1- 9/3/18 = 0.83

Prec(GT(0,2)=1- 18/3/16 = 0.67

The above calculated values prove that the highest precision is GT(0,1)=0.83, therefore, it will be picked by the generalization algorithm.

The generalization using PT for each attribute is practically not possible for a large size of data, therefor, the real-world data is generalized and supressed using tuples instead of individual attributes. Of these systems is datafly system, The system is given the most important field, so it will be generalized, for example D\_O\_B is generalized to the year of birth instead. The next step is counting the number of times of the tuple occurrence. The non-repeated tuples with frequency=0 will be supressed.

Another popular system is μ-Argus, this system categorizes the attributes based on their sensitivity. The values given are: 0 (Not Identifying), 1 (Most Identifying), 2 (More Identifying), and 3 (Identifying) respectively. μ-Argus supress cells instead of supressing the whole tuple, as mentioned in datafly.

# Top-Down Specialization TDS

TDS algorithm was developed to achieve LKC-Privacy on high-dimensional data, the algorithm is also called HDTDS. The idea is starting from the most general value in the taxonomy tree, and then move to the bottom of the tree. The taxonomy tree should be built in advance for each attribute.

This method uses both of generalization and classification as a masking process, determining the best masking process depends on the user needs, some users need precise and deep analysis details while others need general results. In data mining, unmodified results is considered to be the lowest cost in any cost matric.

However, all k-anonymization methods may use either the tuples, the attribute rows, the multi-dimensional or the cells. Fung et al [8] has studied both of cells and multi-dimensional methods, the study showed that less information loss in cell generalization, comparing to multi-dimensional model.

The k-anonymization adopts three different techniques for masking data, the data provider determines the QID before any masking technique takes a place. The masking techniques are generalization, suppression, and discretization. It is essential to define data to categorical and numerical before start anonymization, this classification helps masking process to identify the best technique.

The generalization compromises a taxonomy tree, for example if the data contains person’s address as “Parramatta”, then the taxonomy tree contains Australia 🡪 NSW 🡪 Parramatta, the generalization of the first “cut” is NSW, and the second cut is Australia.

The suppression does not use the taxonomy tree, instead it replaces all occurrence values with a similar value, usually suppression appear as (\*). Some data cannot be generalized, it can be shown or supressed, for example the person’s gender {Male,Female}, such a gender example, it can be hidden by using “Person” that replaces the gender. The notation “Subj” is used when using suppression in algorithms.

The Discretization replaces data values with an interval, which leads giving general range of value. This technique hides the exact value and facilitates the grouping techniques if applied.

The Refinement is used to reduce data anonymity, less data noise, leads to a better refinement. The refinement increases the information gain InfoGain(v), and reduces the anonymity *AnonyLoss(v)*, the trade-off between gain and loss is presented as [9]:

The value 1 was added to avoid division by zero. This equation doesn’t satisfy the form matric to capture the classification, therefore, Shannon’s equation is used for correctness.

Next finding both of InfoGain and AnonyLoss to determine the best generalization for each attribute, this depends on the QID used on each analytics.

Where is the entropy of T(x). To find out the best score for a compressed or generalized table, let us consider table 5 with an extra attribute of “Education”, the education attributes describes the patient’s education level starting from the year 9 – postgraduate studies. Herein we use the TDS model, and the compressed records must start from the root of taxonomy tree. Table 9 shows the

Table 11, Compressed Patient table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Education** | **Sex** | **Work\_hrs** | **Class** | **# of Records** |
| 10th | M | 40 | 20Y0N | 20 |
| 10th | M | 30 | 0Y4N | 4 |
| 9th | M | 30 | 0Y2N | 2 |
| 9th | F | 30 | 0Y4N | 4 |
| 9th | F | 40 | 0Y6N | 6 |
| 8th | F | 30 | 0Y2N | 2 |
| 8th | F | 40 | 0Y2N | 2 |
| **Total:** | | | **20Y20N** | **40** |

To calculate the InfoGain, and InfoLoss for each the table, we start first from the most top generalization, which is ANY\_Edu, for the whole records in the table.

* QID={Eudcation,sex,work\_hrs}
* Number of Records=40
* E(T[ANY\_Edu])=
* E(T[8th])=
* E(T[9th])=
* E(T[10th])=
* InfoGain(ANY\_Edu)= E(T[ANY\_Edu]) – (
* **InfoGain(ANY\_Edu)=1-(0+0+24/40\*0.65)=0.6**

While the InfoGain for the sex is calculated as:

* E(T[ANY\_Sex])=
* E(T[M])=
* E(T[F])=
* **InfoGain(Any\_Sex)=E([Any\_Sex])- (**
* E(T[1-99))=
* E(T[1-40))=
* E(T[40-99))=
* **InfoGain([1-99))=0.39**

Table 12, shows the specialization starts with Education, the highest InfoGain

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Education** | **Sex** | **Work\_hrs** | **Class** | **# of Records** |
| 10th | ANY\_Sex | [1-99) | 20Y4N | 24 |
| 9th | ANY\_Sex | [1-99) | 0Y12N | 12 |
| 8th | ANY\_Sex | [1-99) | 0Y4N | 4 |

The highest InfoGain is (ANY\_Edu), so the specialization for education starts first as shown in table 12, in contrast to AnonyLoss, which shows “Sex” should be generalizaed first, as calculating AnonyLoss uses the following equation:

AnonyLoss(ANY Edu) = A(QID) – A(ANY\_Edu(QID))

The average of AnonyLoss is usually calculated to find out the best generalization and specialization for each attribute. The total results can be determine by calculating the score for each attribute, the

Score(v)=InfoGain / AnonyLoss + 1

For example, Score(ANY\_Edu)=0.0165, and Score(ANY\_Sex)=0.0183, and for [1-99)=0.0136. This can determines that the ANY\_Sex score is the highest.

# MapReduce

MapReduce [9] is a framework for performing data intensive computations in parallel on commodity computers. A MapReduce computation reads input files from a distributed file system which splits the file into multiple chunks. Each chunk is assigned to a mapper which reads the data, performs some computation, and outputs a list of key/value pairs. In the next phase, reducers combine the values belonging to each distinct key according to some function and write the result into an output file. The framework ensures fault-tolerant execution of mappers and reducers while scheduling them in parallel on any machine (node) in the system [10].

Parallel and Distributed computing, for a large size of data, more than 10’s of Terabytes. Since the MapReduce uses split, map, shuffle and reduce, therefore, any practical security solution should take these main processes in the consideration. Any tweaking in the available algorithms should consider the milestones of the scale-up efficiency and the data privacy [11].

The previous mentioned algorithms and analytics models are still valid for MapReduce, k-anonymity, l-diversity, and others can be used with few modifications. These modifications are essential to provide a crucial compatibility to MapReduce drivers, and clusters performance. However, most of these anonymization models are applied after the MapReduce phase is completed.

Many Models have been developed especially for MapReduce such as Two Phase TDS or TPDS [12], Airavat [13], PINQ and GUPT [14].

## Airavat

Airavat is a novel MapReduce security and privacy framework. It provides Mandatory Access Control MAC for Mappers, by enforcing MAC on both sides of MapReduce processes and Mapper output. Airavat follows the Analytics process step by step, starting from the user Mapper query, determining if the required keyword is single or multiple. Enforcing the MAC policy during the MAP-Reduce processing, and finally adding noise to the keyword and comparing it with the key output.

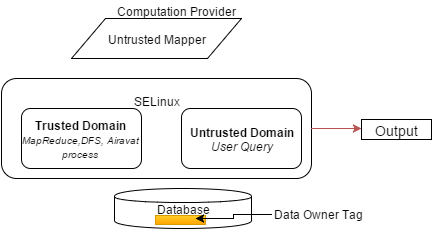


Figure 2, High-Level architecture of Airavat

Airavat allows the execution of trusted and untrusted MapReduce on sensitive data, by enforcing the data owner policy using “Declassify Tag”. Many security features have been add to Airavat framework, to prevent different side-channel attacks like state attack, privacy attack and time attack.

Airavat architecture divides the procedures in to three parties, these are the data owner, the user or mapper, and the computation framework (Airavat). The user first plans his/her code for Map and Reduce. Two types of users are pre-defined in Airavat, trusted and untrusted, the untrusted user keyword is (noisy) hidden on output, and the untrusted user is not allowed view it. Also, the untrusted user is prohibited from executing all queries, queries like “list all” is not permitted. On contrast, trusted user is permitted to use any queries. Airavat can’t confine the keyword as a sensitive value or not, keywords are usually strings, and determining the string related attribute is not provided by Airavat model.

MapReduce Files, DFS, SELinux files have been modified in Airavat model, the modification is necessary to accommodate the data owner declassification flag or DF, the flag is added to the database, with a permission possibility to keep or remove this flag by Airavat. The output result or key/value pair are manipulated, Airavat uses differential privacy to create noise by using Laplace equations. Airavat sorts trusted user keys prior the output, so they don’t output in the same input sequence, so the attacker can’t use the key order to leak information.

SELinux is divided into two domains, one is untrusted for user code, and the other is trusted for Airavat files, MapReduce, and DFS files. As shown in figure 2, both of these domains available in SELinux environment, the domain access and processes are controlled by MAC.

Airavat suffer from some limitations in confining the untrusted code, this is because of the keyword mapping difficulties, as described before. MAC is only used to control the user’s access and processes, while MAC doesn’t provide any mechanism for interacting with differential privacy or choosing sensitive attributes. The access control is not implemented to fully distinguish different levels of access, hence, the user is categorized as trusted or untrusted, which is two levels of privileges.

## GUPT

GUPT was proposed to reduce the analytics complexity for an average programmer. GUPT considers that the data owner and the service provider are trusted, while the analyst is untrusted. To gain this aim, GUPT framework is divided into three blocks, these are; data set manager, computation manager, and isolation execution chambers. Dataset manager is a database to maintain the data privacy, while computation manager handles the computations process by transferring data from the dataset manager, to the appropriate instances. Finally, the isolation execution chambers isolate and prevent any malicious behaviour.

GUPT uses the differential privacy to protect the final output. It also use the Laplace to investigate the noise level and accuracy. The analyser query is evaluated on a smaller data blocks, since the search method uses small blocks and aging sensitivity. This means, the older age data are assigned by less sensitive values.

The optimal block size can be attained by finding a trade-off between the error and noise, which reduces the final error to a large extent. The block size varies from one query to another, this can be presented by , where n is the size of dataset, and is a parameter to be ascertained.

GUPT uses Mandatory Access Control framework MAC, to secure communication between instances, and each process performs in its designated space. AppArmor is used with SELinux as a sandbox to manage MAC. The computation manager is split into server-client, the client allows GUPT to disable all network activates for the untrusted computation.

# Problems with the current Analytics Models

Current analytics models suffer from security authorization and privileges, some of the models have partially implemented MAC to control the user’s processes and to stop leakage between these processes. The leakage can be technically stopped using other techniques such as creating a separate domain for users code, as in Airavat model, users can be redirected to a less secure server to run their code, the server can be isolated from the network. Another technique is using other access control models such as Role-Based Access Control RBAC, which provides more powerful functions than MAC.

The current models don’t provide any authorization method to distinguish different levels of access, Big Data contains a massive size of objects, these objects vary between database schemas, and files. In consequence of that many users access the database, for read, write and analytics. Users are belong to many domains and different organizations. Also, the database is collected from different owners. Therefore, accessing Big Data objects should be controlled disparity methods. This implies multi-level of access for each user. For example suppose a user U is intending to access objects O={O1,O2,…..Oi}, only objects Ṑ={Oj,Oj+1,….On} are owned by U, the objects number Ṑ ≤ O. Current analytics frameworks haven’t identified any definition to deal with ownership, co-ownership, partnership, and others.

Differential privacy methods defines QID, K, L, and C for sensitivity, it uses Laplace noise or generalization, discretization and suppression to hide the QID output. The models doesn’t show any technique to find out the QID, C, K, and L, most of these variables are defined without referring back to the user, who inquire this data. There is no sense of applying generalization, for example, on the data owner, also, data co-owner, or partner should be given more data than customer, for instance.

A need for an access control model to provide an access discrimination between objects is essential. The access control should be available ad integrated with the Federation service. The cloud service provider does not need to keep a record of user’s access levels and privileges. A fine grained access control data should be stored with the authentication organization. The access control model should be integrated with the other access control models for the cloud resources. The access control model should be able to 1) Manage the access permission for Analytics. 2) Verify the permit/deny MapReduce query. 3) Controls the processes Leakage during MapReduce. 4) Supports identifying (QID,K,L,C) or noise level.

RBAC model is strongly elected to provide part of the above mentioned requirements. SELinux implements Multi Level Security, including MAC, and RBAC. SELinux Type Enforcement TE model differs from traditional mode. SELinux implements TE under the layer of RBAC, which controls users processes with a highly granular access control[15]. RBAC is more popular than MAC, it is being implemented in almost all applications operating systems, and this provides access permissions with authentications, and controls the processes leakage during MapReduce. However, RBAC is unable to decide the permission for MapReduce query, and unable support the decision of identifying (QID,K,L,C) and noise.

# My Contribution

We describe three analytics related parties in Big Data; these are the data owner, the service provider, and the user analyser. Based on the previously mentioned analytics models, we introduce SLB-RBAC model that is able to protect side attack including state, privacy and timing attacks. The model uses RBAC core parallel with attribute Sensitivity Level Base (SLB), to manage the followings: 1) Manage the access permission for Analytics. 2) Verify the permit/deny MapReduce query. 3) Controls the processes Leakage during MapReduce. 4) Supports identifying (QID,K,L,C). the model SLB interacts with RBAC, throughout mapping both models to investigate the above four requirements. SLB-RBAC uses the MDTDS model to anonymize data.

## Preliminary

SLB-RBAC will be applied for K-Anonymity first, then it will be scaled up to the other anonymization methods. SLB-RBAC is manages the relationship between data owners and system analysers (users). Data owners may transfer the authorization and authentication to any related federation service, for example the cloud service provider. Therefore, the word federation is meant to be the authorization and authentication service, whether it is a part of the data owner belonging or a third party support. The concept of SLB-RBAC is mapping the RBAC roles in the data owner side, so the MapReduce can use the mapped notations in the cloud side. The data owner pre-defines security policies by classifying the data attributes, and assigning general policy.

## SLB-RBAC Method

The SLB method mandates to find the number of QIDs that needed to be generalized and the level of generalization. Users define subset of attributes as Quasi Identifiers, while defining other subset of attributes as sensitive attributes, the SLB task is deciding the number and level of generalized QID. The following steps are applied during the anonymization process:

Step 1: The Data owner chooses the object QIDs, and the object sensitive attributes.

Step 2: Data owner chooses the object Obsolescence value

Step3: Apply the TDS method and find the highest InfoGain, and score in all QIDs

Step 4: Use Sensitivity Level to determine the number of used QIDs in generalization, and starting from the highest score QIDs.

Step 5: Use the Access level to determine the generalization level.

## Definitions

**Definition 1**: Sensitivity level (SL) is finding a scale of prominence for a subset of attributes subject to the privacy constraint.

**Definitions 2**: The data link attack is more possible on the increase number of QIDs. Therefore, more QIDs lead to higher Sensitivity Level (SL).

# Equations

### The Sensitivity Level Equation

The SL is calculated first to determine the number of QIDs chosen in generalization and specialization, consider this as a pre-generalization process that determines how much the sensitive attributes are important. For instance if the data owners chooses (marital status) a sensitive attribute, and (job,address,background,visa status) are Quasi Identifiers. The SL is calculated first, to determine the level of sensitivity for (marital status), suppose that SL=0.75, therefore, 4 X 0.75 = 3, which means 3 QIDs will be considered for this query only. The chosen 3 QIDS will be determined after calculating the highest score for each QID. Let us consider that (job,address,visa status) are the highest scores, and (job) is the highest attribute between the four of QIDs, in this case, job will not be generalized, while both of address and visa status will be generalized according to the RBAC level.

The data owner determines the sensitive attributes, and QIDs for each object, while the SL is determined by three factors, these are: The ownership level (*ol*). 2) The object Age Ṑ.

**…….(1)**

The above factors mandate SLB to decide the SL, and as a result evaluates the chosen percentage of QID, the percentage range falls between [0% - 100%].

The number of QIDs used in generalization is calculated by using QID(SL).

***QID(SL) = SL X QID* …………………………………………………………..(2)**

The sensitivity level determines the number of chosen QID’s during the anonymization process. Higher sensitivity level conducts to a higher number of QID’s used in anonymization.

### The Generalization Level Equation

The GL determines the generalization level in each *QID(SL),* the method invokes the InfoGain value for each attribute. Three main levels of access are assigned to each ownership level; these are: High, Medium, and Low. For example a user with ol=”Co-Owner” may access with either “High”, “Medium” or even “Low”. The access level truncates the taxonomy tree “cut”, starting from the parent to the child(v).

The following equation determines the cut level based on the InfoGain and access level:

**……………………………………(3)**

Where

Figure 3, and 4 illustrate the taxonomy tree for job, and suburb. The GL is determined based on the access level, for example if the GL(job)=0.80 for a “sales manager”, and QID=4, means 0.8 X 4=3.2, mapping 3.2 on a taxonomy tree attains “Manager”. This is derived from GL=0.25 🡪(Any=1), G=0.5 🡪(Employee=2), and finally GL=0.75 🡪(Manager=3)

## 

Figure 3, Taxonomy Tree for JOB

## 

Figure 4, Taxonomy Tree for Suburb

## Ownership Level (*ol*)

The ownership level is the main factor in measuring the sensitivity level, it depends on the user’s level of access. Each ownership value is given an integer number (n), where n={0,1,2,3,4}, the given values are: Owner (n=4), Co-Owner (n=3), Partner (n=2), Contractor (n=1), Public (n=0).

For example a table T1 with 5 QIDs is more sensitive than table T2 with 2 QIDs, if T1 was accessed by a user, then the system has to define the number of QID’s that are needed to be processed. Depending on the user ownership, and using table 13, ownership level for the user can be calculated using the following equation:

Where n is the ownership value

Table 13 denote the sensitivity level for any access. For example if a user of an ownership level “Partner (2)” has enquired a list of data with QID=9. Then the *ol* is:

While QID=9, then 9 X 0.78 = 7. Instead of using 9 QIDs, only 7 QIDs are used to proceed the next step.

Table 13, the sensitivity levels used in SLB-RBAC for different numbers of QID

|  |  |  |
| --- | --- | --- |
| Ownership (n) | QID | Allowed Specializing QID No. |
| Public (0) | 2 | 0 |
| Contractor (1) | 2 | 1 |
| Partner (2) | 2 | 1 |
| CO-OWNER (3) | 2 | 2 |
|  | | |
| Public (0) | 3 | 0 |
| Contractor (1) | 3 | 1 |
| Partner (2) | 3 | 2 |
| CO-OWNER (3) | 3 | 3 |
|  | | |
| Public (0) | 4 | 1 |
| Contractor (1) | 4 | 2 |
| Partner (2) | 4 | 3 |
| CO-OWNER (3) | 4 | 4 |
|  | | |
| Public (0) | 5 | 2 |
| Contractor (1) | 5 | 3 |
| Partner (2) | 5 | 4 |
| CO-OWNER (3) | 5 | 5 |
|  | | |
| Public (0) | 6 | 3 |
| Contractor (1) | 6 | 4 |
| Partner (2) | 6 | 5 |
| CO-OWNER (3) | 6 | 6 |

## The Object Aging Sensitivity (Ṑ)

The object age sensitivity (Ṑ) is reversal, whereas the older objects carry less sensitive information, the sensitivity changes in an exponential relationship of y years. The age reads from the archive data to evaluate the object age. The object age sensitivity can be pre-determined by the data owner. Eventually different objects contain different values, lean to different ages. Some objects data may be ignored after few years, while others may last for infinitive years. Each created object should be provided with aging data period. The owner may assign the obsolescence value as obs= {5,10,20,30,……} of years. Let us consider the obs=10 years for a group of objects. Table 14 shows the calculated percentage for each one of these objects.

Table 14, the linear sensitive percentage for the Obsolescence=10

|  |  |
| --- | --- |
| **Aging Sensitivity Ṑ (%)** | **Object Age (** |
| 10 | 10 |
| 20 | 20 |
| 30 | 30 |
| 40 | 40 |
| … | … |
| 100 | 100 |

## Calculating the SLB Example

Suppose the following Customers table, as shown in table 18, is located in a social website. The data owner needs to move the database to the cloud for analytics, but at the same time, he needs to hide some private information. The owner has considered that CONTACTNAM, and PHONE must be completely hidden, while CONTACTTIT, GENDER, POSTCODE, SUBURB,isMarried are sensitive information and are supposed to be protect from any side link attack.

Table 18, subset records of Customers table with 4 QIDs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| CONTACTNAM | CONTACTTIT | GENDER | POSTCODE | SUBURB | MOBILE | isMarried |
| Maria Anders | Sales Representative | F | 2150 | Parramatta | 405885514 | Yes |
| Ana Trujillo | Owner | F | 2150 | Harris Park | 456823599 | No |
| Frédérique Citeaux | Marketing Manager | M | 2141 | Lidcombe | 433699987 | Yes |
| Patricio Simpson | Sales Agent | M | 2060 | North Sydney | 406542215 | No |
| Francisco Chang | Marketing Manager | M | 2340 | Kingswood | 421547454 | Yes |

To apply SLB-RBAC let us assume the following scenario. A user with ownership “Partner”, and access level “High”. The data owner has determined the age obs=2 years, since the marital status may changes within a short period of time. Consider the table age is 3.5 years old.

The user has used PIG / MapReduce language to view all users who are still single. In this case we can apply LKC-Privacy to gain the required anonymity. Following the previous mentioned steps will rectify and refine the final anonymization for the above table.

Table 19 shows the QIDs and Class after hiding the sensitive attributes. The table is created by counting the number of records for each similar tuple.

Table 19, repeated tuples with QID=4, and class “isMarried”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CONTACTTIT** | **GENDER** | **POSTALCODE** | **SUBURB** | **isMarried** | **# of Records** |
| Sales Representative | F | 2150 | Parramatta | 12Y3N | 15 |
| Sales Representative | M | 2150 | Parramatta | 2Y5N | 7 |
| Owner | F | 2150 | Harris Park | 3Y1N | 4 |
| Owner | M | 2150 | Harris Park | 1Y1N | 2 |
| Order Administrator | F | 2135 | Starthfield | 11Y1N | 12 |
| Marketing Manager | M | 2141 | Lidcombe | 10Y2N | 12 |
| Accounting Manager | F | 2060 | N. Sydney | 10Y0N | 10 |
| Sales Agent | M | 2064 | Artarmon | 2Y10N | 12 |
| Sales Agent | F | 2064 | Artarmon | 1Y0N | 1 |
|  |  |  |  | **52Y23N** | **75** |

Step 1: The Data owner chooses the object QID, and sensitive attributes:

* Sensitive Attributes={CONTACTNAM, MOBILE}
* QID={ CONTACTTIT,SEX,POSTCODE,SUBURB,isMarried}
* Let us consider isMarried is the class C with %50 (Yes or No)

Step 2: Data owner chooses the object obs.=2

Step3: Apply the TDS method and find the highest score in all QIDs

I(any\_job)=

I(self\_employed)=

I(Employee)=

InfoGain(any\_job)=0.889 -

AnonyLoss(any\_job) = (75 – 6) / 1 = 69

Score(any\_job)= 0.092/(69+1)= 0.001317

Table 20 shows the results of calculating InfoGain, AnonyLoss, and scores for each attribute

Table 20, the results of infoGain, anonyLoss, and Score

|  |  |  |  |
| --- | --- | --- | --- |
| **Candidate** | **infoGain** | **anonyLoss** | **Score** |
| Any\_Job | 0.092211 | 69 | 0.001317 |
| Any\_Gender | 0.325026 | 43 | 0.007387 |
| Any\_Postcode | -0.08429 | 62 | 0.00134 |
| Any\_Suburb | -0.08429 | 62 | 0.00134 |

The (Any\_Gender) is the top most scores, and the best refinement is applied, therefore any gender will not be generalized.

Step 4: Use Sensitivity Level to determine the number of used QIDs, and starting from the highest score QIDs.

**=**

As a result the total number of generalized QIDs = 4 X 0.29 = 1.16 ≈ 1

The SL value indicates that at one QID attribute must be generalized, while the rest of QIDs are refined to best level. The highest InfoGain will be chosen for generalization, which is Any\_Job in this example. The generalized level for access level “High” is calculated as:

Step 5: Use the Access level to determine the generalization level.

The GL value indicates the cut in taxonomy tree can be found by 0.822 X 4 = 3

Referring to figure 3, the third cut level is optioned between “Manager” or “Professional”.

Table 21, shows the final masking and refinement result for Partner-High-age=3.5

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CONTACTTIT** | **GENDER** | **POSTALCODE** | **SUBURB** | **isMarried** | **# of Records** |
| Professional | F | 2150 | Parramatta | 12Y3N | 15 |
| Professional | M | 2150 | Parramatta | 2Y5N | 7 |
| Self Employed | F | 2150 | Harris Park | 3Y1N | 4 |
| Self Employed | M | 2150 | Harris Park | 1Y1N | 2 |
| Professional | F | 2135 | Starthfield | 11Y1N | 12 |
| Professional | M | 2141 | Lidcombe | 10Y2N | 12 |
| Professional | F | 2060 | N. Sydney | 10Y0N | 10 |
| Professional | M | 2064 | Artarmon | 2Y10N | 12 |
| Professional | F | 2064 | Artarmon | 1Y0N | 1 |
|  |  |  |  | **52Y23N** | **75** |

Consider the same previous data with user “Public-Low” access, and database age=6 months, it’s expected this increases the SL, and decreases the GL. After using the previous equations:

= 0.03 , OL=1, SL=3.34, GL=0.27

Table 22 shows the final masking after refinement for Public-Low-Age=6 months

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CONTACTTIT** | **GENDER** | **POSTALCODE** | **SUBURB** | **isMarried** | **# of Records** |
| Any\_job | F | \*\*\*\* | Any\_suburb | 37Y5N | 42 |
| Any\_job | M | \*\*\*\* | Any\_suburb | 15Y18N | 33 |
|  |  |  |  | **52Y23N** | **75** |

## SLB specifications

HDTDS is proposed to achieve LKC-Privacy on high-dimensional data. HDTDS uses the InfoGain, and AnonyLoss to refine the anonymization, this can be done by refining the generalization, suppression and discretization. The anonymization process requires generalization based on taxonomy tree, so grouping the records will be easier, and as a result values anonymization will be investigated.

SLB segregates the user queries into two parts, the keywords, and the requested attributes. Since the user query is usually contributed by scripts rather than sql commands, it is essential to control the keywords as well as the requested attributes. SLB will aim toward finding MyBenchmark in queries [16], the benchmark is essential to measure any used algorithm in analytics performance. Proposed algorithms keen to protect against data leakage in data analytics. On the other side, data analytics computational cost is high, hence, speed and performance are essential in big data. A trade-off between security and performance should be consider on developing any security algorithms. One time scan for any generalization process is fair enough to increase the processing time in big data, less scanning time, will result a better performance.

Big Data adopts MapReduce model for data analytics. Therefore, my future work in SLB model will focus on using highly scalable two-phase TDS approach using MapReduce [12]. Analytics results of Data sets are partitioned and anonymised in the first phase on parallel method, the results is so called intermediate, then the intermediate is merged in reduce and further anonymised to produce consistent  k-anonymous data sets in a second phase.

# RBAC Mapping

The SLB-RBAC model maps the users RBAC roles to the analytics levels, organizations delegate other organizations or personnel to analyse their own data. The delegation can distinguish between them, partners may have higher privileges than customers, while co-owners may able to read less anonymised data. Data Owner RBAC is supposed to provide the data provider RBAC with three main variables; these are the user unique id, the ownership level the user level. These variables can be transferred within an xml file, and using any assertion method such as SAML, or XACML.

This model divides the each ownership level into three user’s levels: high, medium, and low; each level measures the anonymity level. The KLC-anonymity uses the scores values by dividing InfoGain on AnonyLoss, as explained before. The user level ignores ownership class, on calculating scores, for example, co-owner with medium level is calculated with the same score equation for the partner.

Shao et al [17] has developed a java based RBAC that supports XACML, the system is able to map between different organization and the service provider. The service provider categorizes access into levels, this means if a user role “Business Manager” was authenticated to access any resources over the cloud from a company A, while another user was authenticated from company B, he also carries the same role “Business Manager”. Both companies attempt to access the same resources in the same provider, meanwhile each user carries two levels of customer roles, level one customer role, and level two customer role.

Shao model is sophisticated enough to compensate the users access to data resources over the cloud, our focus is mainly on mapping from any RBAC model, to a naive model that is able to support a multi-level of data analytics. SLB-RBAC provides a practical tool embedded in the authorised federation service, the federation service must be an accredited registered federation with the service provider. The model provides SAML assertion and server between federation and service provider. The user, who wishes to participate in data analytics, attempts to access the data, the following steps are applied for authorization and authentication:

Step 1: the data service provider direct the user to the right federation service, the right federation service is approved by the data owner and cloud service provider.

Step 2: the user is authenticated using RBAC system.

Step 3: RBAC verifies the user role for system analytics.

Step 4: RBAC federation contains the following business roles:

Table 18, RBAC federation and business roles

|  |  |  |
| --- | --- | --- |
| **SLB OWNERSHIP** | **RBAC LEVEL** | **RBAC ROLE NAME** |
| Owner | High | OWN\_HIGH |
| Owner | Medium | OWN\_MED |
| Owner | Low | OWN\_LOW |
| Co-Owner | High | COOWN\_HIGH |
| Co-Owner | Medium | COOW\_MED |
| Co-Owner | Low | COOW\_LOW |
| Partner | High09 | PARTN\_HIGH |
| Partner | Medium | PARTN\_MED |
| Partner | Low | PARTN\_LOW |
| Contractor | High | CONTR\_HIGH |
| Contractor | Medium | CONTR\_MED |
| Contractor | Low | CONTR\_LOW |
| Public | High | PUB\_HIGH |
| Public | Medium | PUB\_MED |
| Public | Low | PUB\_LOW |

Step 5: an xml file will be sent to the service provider using SAML server

Step 6: RBAC service provider reads the xml file and provides the user with the correct match RBAC business role. The RBAC business roles are similar to the federation business roles.

Step 7: the above business roles are divided into two separate groups <ol, ul>, notated for two different levels, the first ol value interacts with sensitivity calculation, while the ul value interacts with score calculations.

# SLB-RBAC Framework

## Security Assertion Markup Language (SAML)

The SLB framework is implemented between the data owner, the cloud provider, and the users who wish to access this data to perform any analytics tasks. The framework is implemented and inherited within RBAC access control, users are given an access level for authentication and authorization.

SAML is an XML base single sign on (sso) standard, which provides authentication and authorization mechanism, with an interoperability between different security services in distributed environments. SSO invokes an ease of user access over the net, for different web services. Three main objects are involved in SAML procedures, these are: Clients, Identity Provider, and Federated service [18].

SAML standard can be implemented in different scenarios, this depends on the business needs and limitations. However, all scenarios follow close similar procedures. In the mean time we will look at the most prominent scenario that initiates SLB framework in analytics. SAML starts with an initiation request by users, who initiates their request to the idP or directly to the service provider. Nevertheless, SP must redirect the user request to the idP for authentication. The request is formatted in assertion notations, which is a set of XML groups that get bind using POST, GET and / or by SOAP server [19][20].

It’s important to distinguish between two binding methods, these are; front-channel bind, and back-channel bind; the front-channel is more popular binding method that is triggered using the web browsers, between client and server at the other end. This conventional method of web communication implements the standard security of TLS/SSL, digital signature, public and private keys, over TCP/443 [21].

The second binding method is usually server-server method, its notion applies two servers communicate with each other, and without human interaction, by initiating connection through a pre-defined port. The first method applies POST and GET binding, while the second method applies SOAP, or reverse SOAP. SOAP is used parallel with artifacts. The communications procedures start by the artifacts generator, which transfer the source ID, a reference, and a message. This artifacts is transferred by using the front-channel method, using POST or GET. The SP verifies the message, and recognizes that it is an artifact, so it generates an XML request known as <ArtifactResolve>. This request is transferred to the idP using SOAP synchronization. The idP generates an XML <response> and sends it back to the SP, the response transfer method can include any of, SOAP, POST, or GET[22].

The idP builds a SAML XML response which contains, assertion, and attributes. The HTML form hides the XML codes, as hidden values, and transfer it to the SP. The HTML form is usually submitted automatically using a script, in order to save the user’s time. The response might be sent using POST or SOAP only.

Figure 5, illustrates the steps used to authenticate user, and tokens transmitted between idP and SP.

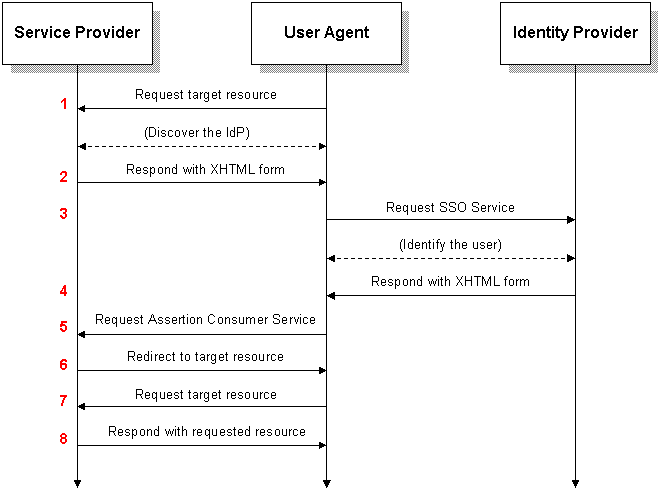


Figure 5, SAML steps

## SAML and XML

SAML adopts XML files in different formats, developers use many methods to read the tokens, this flexible option enables developers to insert additional variables and values. Figure 6 shows an example of XML response assertion for SAML 2.0.The example divides the XML file into 8 sections these are: response ID, Issuer ID, Status (success or Fail), Assertion ID, Signature key, Conditions, Authentication statement, and Attribute Statement. The last section contains unlimited names and values of attributes, developers use this section to pass any authorization attributes and values.

1. **<samlp:Response ID="\_257f9d9e9fa14962c0803903a6ccad931245264310738"**

IssueInstant="2009-06-17T18:45:10.738Z" Version="2.0">

1. **<saml:Issuer Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">**

https://www.salesforce.com

**</saml:Issuer>**

1. **<samlp:Status>**

<samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>

**</samlp:Status>**

1. **<saml:Assertion ID="\_3c39bc0fe7b13769cab2f6f45eba801b1245264310738"**

IssueInstant="2009-06-17T18:45:10.738Z" Version="2.0">

<saml:Issuer Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">

https://www.salesforce.com

</saml:Issuer>

1. **<saml:Signature>**

…………………….

**</saml:Signature>**

1. **<saml:Conditions NotBefore="2013-06-17T18:45:10.738Z"**

NotOnOrAfter="2013-06-17T18:50:10.738Z">

<saml:AudienceRestriction>

<saml:Audience>https://saml.salesforce.com</saml:Audience>

</saml:AudienceRestriction>

**</saml:Conditions>**

1. **<saml:AuthnStatement AuthnInstant="2013-06-17T18:45:10.738Z">**

<saml:AuthnContext>

<saml:AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:unspecified

</saml:AuthnContextClassRef>

</saml:AuthnContext>

**</saml:AuthnStatement>**

**8. <saml:AttributeStatement>**

<saml:Attribute Name="portal\_id">

<saml:AttributeValue xsi:type="xs:anyType">060D00000000SHZ

</saml:AttributeValue>

</saml:Attribute>

<saml:Attribute Name="organization\_id">

<saml:AttributeValue xsi:type="xs:anyType">00DD0000000F7L5

</saml:AttributeValue>

</saml:Attribute>

<saml:Attribute Name="logouturl"

NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri">

<saml:AttributeValue xsi:type="xs:string">

http://www.salesforce.com/security/del\_auth/SsoLogoutPage.html

</saml:AttributeValue>

</saml:Attribute>

</saml:AttributeStatement>

</saml:Assertion>

Figure 6, XML response example for SAML 2.0 [23]

In SLB we need to inform the RBAC system with the ownership and access levels, so the SLB algorithms calculates the SL, GL, and OL. This can be applied by embedding one more attribute with the XML response file, this can be embedded in the Attribute section. The insertion can be formatted similar to this example in Figure 7:

**<saml:Attribute Name=”analytics”>**

<saml:AttributeValue xsi:type=”xs:anyType”>

PARTN\_MED

</saml:AttributeValue>

**</saml:Attribute>**

Figure 7, Analytics insertion in the Attributes section for XML response

## SLB and SAML

After introducing SAML procedures in authentication and authorization, we mentioned that accessing the data over the cloud for analytics can recall SAML technology to facilitate the process over the cloud. SLB uses the current available technology to authorize analytics process for users. The procedures are straightforward with SAML. Users need to setup SAML server for idP and/or Federation, both of idP and Federation must be approved by the service provider (SP).

The steps start from the federation service, who uses RBAC to create 15 business roles, as explained before in table 18. The federation service needs to provide SOAP server with the analytics attribute, as shown in figure 7. Data owner assigns ownership and access level for any user requesting analytics, the user is given an authentication username and password, and assigned in one of the 15 available business roles.

Since hadoop is a Linux base program, and all hadoop daemon run in Linux environment, there is a need to control and prevent these services from any intrusions or attacks. Hadoop processes are distributed between NameNode servers, DataNode servers, and HDFS servers. Also hadoop use some directories to process the job, and place the logs, and hadoop can’t run without giving the proper security permission for these directories.

SLB requires pre-definitions for some users and group in Linux shell, these groups represent the roles in RBAC. For example Solaris 8 or higher use RBAC to control the entire processes in all UNIX shells, the idea is creating 15 group of roles for hadoop analytics. As mentioned before in table 18, the roles are: {OWN\_HIGH,OWN\_MED,OWN\_LOW,COOWN\_HIGH,COOW\_MED,COOW\_LOW,PARTN\_HIGH,PARTN\_MED,PARTN\_LOW,CONTR\_HIGH,CONTR\_MED,CONTR\_LOW,PUB\_HIGH,PUB\_MED,PUB\_LOW}.

The roles can be created using the shell command “ root#groupadd OWN\_HIGH”, each group may contain a number of users, say 5 users for each group, the users are also predefined and ready for any analytics request. The users can be created using shell command “root#useradd –m –g **user\_own\_high\_1** OWN\_HIGH” . A total of 75 users can be created for SLB-RBAC. However, more or less users can be created in each group, this depends on the quantity of users access at a time, each user account is given a password and assigned to any external query by any user. The user account is mapped to any user if SAML server provides a valid XML response, XML file must contain the “analytics” attribute supported by the required role.

The procedures, of mapping any external user to the pre-defined users, is illustrated in figure 9. A script is used to decide the correct group and username for the request. Hence, the real user identity must be recorded after mapping to usernames or groups.

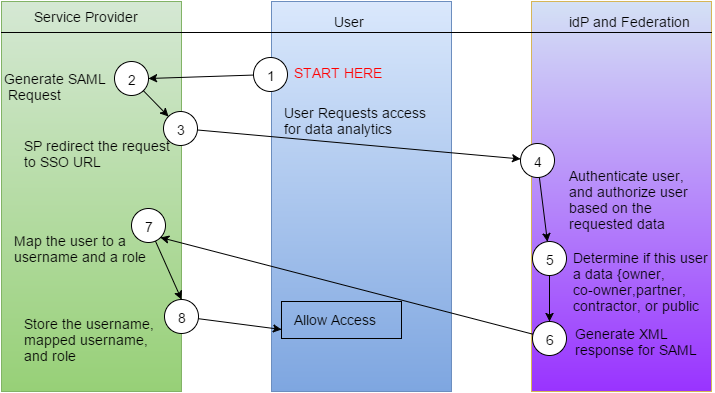


Figure 8, SAML Request and Response and SLB access.

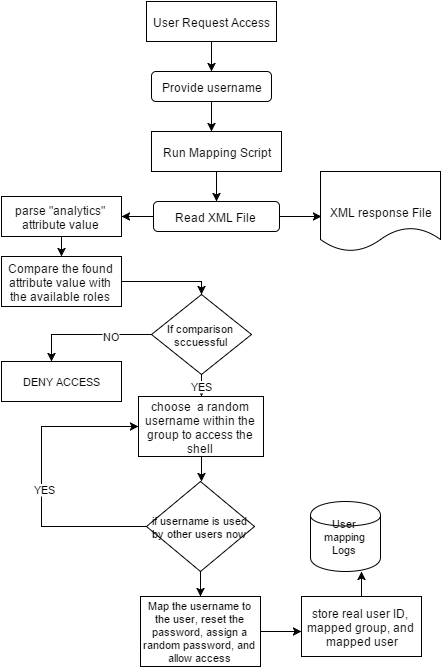


Figure 9, illustrates the algorithm used for mapping users and groups (roles).

# SLB Experiment

I conducted intensive experiment to evaluate the impact of SLB-RBAC, the experiment is conducted in Solaris 11 operating system and hadoop version 2.5.2. Hadoop is installed and configured on a single-node cluster, which imposes the creation of one pool with five Solaris zones, the hadoop cluster building block zones are; name-node, secondary-name-node, data-node1, data-node2, and data-node3 .

Five virtual network interfaces are also used in this architecture to provide one vnic for each zone. The five zones are built with Hadoop File System HDFS, instead of ZFS. This infrastructure is necessary to run high level languages such as PIG and HIVE. I downloaded PIG LATIN and run all experiments through “grunt” shell. The hardware resources that run the single-node cluster are 6 GB RAM, with 1.8 GHz Intel Celeron CPU. A virtual Oracle Box is used for Solaris 11.

User Definition Function (UDF) is used to implement PIG commands, UDF is a way to specify custom processing by users or data owners. UDF can be implemented in three languages, Python, Java, and JavaScript, which enables developers to embed PIG in web interfaces. Java Eval is used in this experiment, which is the most common function in UDF.

Three PIG queries are executed with the help of XML definitions. In this stage user definition is stored in XML files and protected in a secure hidden directory. Also, one Java program is used to generalize the data attributes. A database csv file with 100,000 records as shown in table 18, is used to evaluate the SLB generalization method.

The first PIG query reads the Users.csv file and hides the sensitive attributes:

*REGISTER /GettingStarted\UDF\hide\_sensitive.jar*

*Users = LOAD '/GettingStarted/Input/Users' using PigStorage(',') as (CONTACTNAME:chararray,CONTACTTITLE:chararray,GENDER:chararray,POSTCODE,SUBURB,MOBILE);*

*AllData = FOREACH Users GENERATE CONTACTNAME,CONTACTTITLE,GENDER,POSTCODE,SUBURB,MOBILE;*

*STORE AllData into '/GettingStarted/OutputUser/' using PigStorage(',');*

A java program (hide\_sensitive.jar is used to hide the “CONTACTNAME, and MOBILE

The results appear as shown in table 19.

Table 19, hiding sensitive attributes query 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| CONTACTNAM | CONTACTTIT | GENDER | POSTCODE | SUBURB | MOBILE | isMarried |
| \*\*\*\*\*\*\* | Marketing Manager | M | 2141 | Lidcombe | \*\*\*\*\*\*\*\*\* | Yes |
| \*\*\*\*\*\*\* | Owner | M | 2194 | Campsie | \*\*\*\*\*\*\*\*\* | Yes |
| \*\*\*\*\*\*\* | Owner | F | 2150 | Harris Park | \*\*\*\*\*\*\*\*\* | No |
| \*\*\*\*\*\*\* | Accounting Manager | F | 2170 | Liverpool | \*\*\*\*\*\*\*\*\* | No |
| \*\*\*\*\*\*\* | Sales Representative | F | 2135 | Starthfield | \*\*\*\*\*\*\*\*\* | Yes |

The second query implies reading the previous output, another Java program is used to generalize the data using TDS only.

*REGISTER /GettingStarted\UDF\generalizeTDS.jar*

*Users = LOAD '/GettingStarted/ OutputUser ' using PigStorage(',') as (CONTACTNAME:chararray,CONTACTTITLE:chararray,GENDER:chararray,POSTCODE,SUBURB,MOBILE);*

*AllData2 = FOREACH Users GENERATE CONTACTNAME,CONTACTTITLE,GENDER,POSTCODE,SUBURB,MOBILE;*

*STORE AllData2 into '/GettingStarted/OutputUser/' using PigStorage(',');*

A java program generalizeTDS.jar is used to calculate each attribute’s InfoGain, AnonyLoss, and Score, then generalize all attributes except the “GENDER”, which is the highest score. XML files are used to store the taxonomy tree for CONTACTTITLE, GENDER, and SUBURB. Java program scans the file once and stores it in another file, each tuple is read once, and stored in an empty file, then the second file, then the third, and so on. The program searches the new file to match the records similarity and then to update the number of records. The following algorithm is used in the Java file:

* Open the file users.csv
* Loop all data records
  + Read the tuple
  + Open the file new.csv
    - Loop all data records
    - Match between the read tuple and the available tuples in new.csv
    - If the record match, then increment the number of records
    - If the record doesn’t match, then store the tuple in a new record in the file
    - Next loop
* Next loop

Finally, the third query implies reading the previous data and generalize the data using SLB. The

The third query implies reading the first query output run similar steps as in generalizeTDS.java, then calculate the SL, GL, before inducting any generalization.

The following results are many round experiments with the followings:

1. First round: compare between TDS and SLB where the user access level is (public-low), this showed a longer processing time when using SLB, as shown in diagram 1. The diagram shows the generalization for both of TDS, and SLB with different records.
2. Second round: compare between TDS and SLB where the user access level is (contractor-medium), this showed a close similar processing time when using SLB, as shown in diagram 2.
3. Third round: compare between TDS and SLB where the user access level is (Co-Owner-high), this showed a shorter processing time when using SLB, as shown in diagram 3.

Note that object age was setup to 5 year, while obs=10.

Diagram 1, Comparison between TDS and SLB for public – low access

Diagram 2, Comparison between TDS and SLB for contractor – medium access

Diagram 3, Comparison between TDS and SLB for co-owner – high access

Diagram 4, compares between the three previous access levels

The experiment results are expected, as SLB doesn’t require any extra scan for the data, it uses the some mathematical algorithms before applying any generalization. It was clear that SLB processing time is higher in low level access users, therefore, it is not suitable for public nature data. SLB can be used in data that require more restricted access in the multi-domain environment.

# Conclusion

In this report, SLB-RBAC was experimented to investigate a scalable level of access in the ownership mode and user level mode. The access model is presented by RBAC core functions and roles, the roles are mapped between the federation services and the cloud service provider. The ownership level was divided into five separate levels of owner co-owner partner, contractor, and public. Also the user level was divided into three separate levels of high medium and low. A set of sensitivity factors model were introduced to enhance a better accuracy on choosing the Quasi Identifier, and to scale stages of security access level. RBAC was integrated with generalization methods, the major integrator was the sensitivity factors through using sensitivity categories of top secret, secret, and confidential for each sensitive attribute. Higher sensitive attribute will lead to consistent use of QID, while less sensitive attributes will lead to less QID generalization processes.

Experimenting SLB-RBAC model showed promising results in non-public data nature. The experiment compared between TDS generalization and SLB generalization, and the results showed a minor contrast between the user’s high level access, and the low level access. Users with the low level access need more generalization with some refinement and cut calculation, which consumes longer time that the higher access user.

# 

# References

[1] A. R. Cynthia Dwork, "The Algorithmic Foundations of Differential Privacy," *Theoretical Computer Science,* vol. 9, 2014.

[2] K. W. Benjamin C.M. Fung, Ada Wai-Chee Fu , Philip S. Yu, *Introduction to Privacy-Preserving Data Publishing: Concepts and Techniques*: CRC Press, 2011.

[3] L. Sweeney, "ACHIEVING -ANONYMITY PRIVACY PROTECTION USING GENERALIZATION AND SUPPRESSION," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems,* vol. 10, pp. 571-588, 2002.

[4] Y. X. Rajeev Motwani, "Efficient Algorithms for Masking and Finding Quasi-Identifiers," 2007.

[5] C. Dwork, "Differential Privacy," *Microsoft,* 2010.

[6] A. Machanavajjhala, D. Kifer, J. Abowd, J. Gehrke, and L. Vilhuber, "Privacy: Theory meets Practice on the Map," ed, 2008, pp. 277-286.

[7] H. Samet, *Multidimensional and Metric Data Structure*: Morgan Kaufmann, 2006.

[8] B. C. M. Fung, K. Wang, and P. S. Yu, "Top-down specialization for information and privacy preservation," ed. USA, 2005, pp. 205-216.

[9] B. C. M. Fung, K. Wang, and P. S. Yu, "Anonymizing Classification Data for Privacy Preservation," *Knowledge and Data Engineering, IEEE Transactions on,* vol. 19, 2007.

[10] Q. Tran and H. Sato, "A solution for privacy protection in mapreduce," ed, 2012, pp. 515-520.

[11] X. Z. Xindong Wu, Gong-Qing Wu,Wei Ding, "Data Mining with Big Data," *IEEE,* 2014.

[12] L. T. Y. Xuyun Zhang, Chang Liu, Jinjun Chen,, "A Scalable Two-Phase Top-Down Specialization Approach for Data Anonymization Using MapReduce on Cloud," *IEEE,* 2014.

[13] S. T. S. Indrajit Roy, Ann Kilzer, Vitaly Shmatikov, Emmett Witchel, "Airavat: Security and Privacy for MapReduce," *The University of Texas at Austin,* 2010.

[14] P. Mohan, A. Thakurta, E. Shi, D. Song, and D. Culler, "GUPT: privacy preserving data analysis made easy," in *SIGMOD '12*, K. S. Candan, uk, Y. Chen, R. Snodgrass, L. Gravano, and A. Fuxman, Eds., ed: ACM, 2012, pp. 349-360.

[15] R. Nimbalkar, P. Patel, and B. Meshram, "Advanced Linux Security," *Editorial Board,* p. 7, 2013.

[16] E. Lo, N. Cheng, W. Lin, W.-K. Hon, and B. Choi, "MyBenchmark: generating databases for query workloads," *The International Journal on Very Large Data Bases,* vol. 23, pp. 895-913, 2014.

[17] S. Li, Q. Aiming, Z. Xiaoling, and Z. Jinzhong, "Improvement and Implementation of RBAC Access Control Model," ed, 2012, pp. 110-115.

[18] O. Org, "Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0," p. 46, 15 March 200 2005.

[19] Venkadesh.M and A. C. S. Dr, "Study on Improving Web Security using SAML Token," *International Journal of Computer Technology and Applications,* vol. 04, p. 917, 2013.

[20] J. Paschoud, "SHIBBOLETH AND SAML: AT LAST, A VIABLE GLOBAL STANDARD FOR RESOURCE ACCESS MANAGEMENT," *New Review of Information Networking,* vol. 10, pp. 147-160, 2004.

[21] O. Canovas, G. Lopez, and A. F. Gomez-Skarmeta, "A credential conversion service for SAML-based scenarios," *PUBLIC KEY INFRASTRUCTURE, PROCEEDINGS,* vol. 3093, pp. 297-305, 2004.

[22] J. B. Rosenberg, *Securing Web services with WS-Security : demystifying WS-Security, WS-Policy, SAML, XML Signature, and XML Encryption*. Indianapolis, IN: Indianapolis, IN : SAMS, 2004.

[23] S. Force. (2010). *Example SAML Assertions*. Available: https://help.salesforce.com/apex/HTViewHelpDoc?id=sso\_saml\_assertion\_examples.htm&language=en