Analysis of Fine-Grained Access Control in Database

Vaibhav N. Dhage
Department of Computer Science & Engg. H.V.P.M. C.O.E.T., Amravati, SGBA University.

R. R. Shelke
Assistant Professor,Department of Computer Science & Engg H.V.P.M. C.O.E.T., Amravati SGBA University.

ABSTRACT

The access control decision is enforced by a mechanism of security policy. Fine Grained Access Control (FGAC) is one of the ways to ensure data security in database. Nowadays, we store a measurable portion of that data in the relational database management systems (RDBMS). Access control is one of the cornerstones of any Information Security Policy. The granularity of such access control can be on different levels, like on directories or folder level, database level, table level, and even on individual record (tuple) and data field level.

In this paper we described the introduction of Access Control Mechanisms of fine-grained access control in databases which determines whether access to a resource is permitted. We also included the study of Oracle's Virtual Private Database. Different models exist for providing access control at Database level is our proposed work.

General Terms

Architecture, Security, Policy.

Keywords

Access Control policy, Data Security, Query, Fine Grained Access Control, DBMS, and VPD.

1. INTRODUCTION

Access control is an integral part of databases and information systems. Granularity of access control refers to the size of individual data items which can be authorized to users. There are many scenarios that demand fine-grained access control:

- For an academic institution's database that stores information about student grades, it may be desired to allow students to see only their own grades. On the other hand, a professor should get access to all grades for a course she has taught.
- For a bank, a customer should be able to query his/her account balance, and no one else's balance. At the same time, a teller should have read access to balances of all accounts but not the addresses of customers corresponding to these balances.
- A Web hosting company may wish to run the HR and Payroll business of other companies. Different companies want different personalization. Some want access to raw data to run business analysis reports that best suit their corporate standards [10].

One strong driving force for fine-grained access control is privacy protection. Another motivation is to move access control from applications to databases. By placing access control at the database level, one can ensure that access control policies are consistently applied to every user and every application [4].

This paper is structured as follows; Section 2 discusses the related work in this area. Section 3 presents about Access Control Mechanisms. Section 4 discusses Oracle's Virtual Private Database. Section 5 explains about the System Architecture. And Section 6 concludes and describes some future work. We ask that authors follow some simple guidelines. In essence, we ask you to make your paper look exactly like this document. The easiest way to do this is simply to download the template, and replace the content with your own material.

2. RELATED WORKS

Fine-grained access control was first introduced as a part of the access control system in INGRES by Stonebraker and Wong (1974), which was implemented by query modification technology [2].

Oracle virtual private database (VPD) also uses query modification to implement FGAC (Oracle Corporation, 2005). VPD supports FGAC through functions written as stored procedures which are associated with a relation [11].

Dr. Nirmal Dagdee and Ruchi Vijaywargiya described Access control methodology for sharing of open and domain confined data using Standard Credentials. This paper was to develop an approach for access control of shared data resources in an environment where open as well as closed domain access control is required. [6]

Neha Sehta, Dr. Suresh Jain introduced A Fine Grained Access Control Model for Relational Databases. A novel access control model is proposed in this work, which provides fine grained access control to shared data to authorized users. In proposed implementation, they have created a set of metatables, to store the data that make up the security policies, registered users and their authorization information [1]. These are some related work.

3. ACCESS CONTROL MECHANISM

Access control determines whether access to a resource is permitted. Permissions and authorization of users or processes are defined according to the policies of the business. An access control policy basically specifies a set of rules that describe the methods in which a client can access a server.

For example MoFAC: Model for Fine-grained Access Control provides tuple level access control. It access to different records in the same file can be treated differently through a subject wanting access to record possessing certain rights, and the object to which access is required, demanding certain rights. Only if the offered rights satisfy the demanded rights, is the access granted.

The basic unit of grouping records is called a Record Group (RG) which is a collection of record IDs and certain other information relevant to records belonging to that RG. The other information may be aspects like what transactions may be executed on records belonging to this RG. As a user logs under a specific role, and requests a transaction to execute, the system checks whether the current role can execute the transaction or not, access may be denied at this point. If the required transaction can be accessed, then the transaction is checked for affected rows, using their IDs various RGs are listed, and each of that RG is checked for whether this

Transaction under this profile can be executed. Only then the final execution takes place or the query is rejected [10].

4. ORACLE'S VIRTUAL PRIVATE DATABASE

Oracle's Virtual Private Database is a feature that combines fine grained access control with secure application context to provide a row level security that presents to each database user only the data that the user should see, based on known characteristics of the user. The main components of the VPD are the application context and the security policy [11].

Fine-grained access control in VPD relies upon dynamic query modification to enforce security policies on the objects with which the policies are associated. Here, query refers to any selection from a table or view, including data updatation, insert or delete statements, or a subquery. Each user should only be allowed to do modifications on the rows that follow the security rules. Based upon these security rules, Oracle generates a predicate clause that transparently appends to the user s SQL statement. This concept is called Virtual Private Database [10].

In Architecture of VPD which as shown in Figure 1, whenever a user connects to the database, the database logon trigger fires, which calls the Context stored procedure to set the defined context. But Policy rules are defined according to the user context. Hence Database also connected to the user context. User issues a query against a policy-linked table, here you can link various policies depending on the select, insert, update, or delete statement. The policy further indicates which function to call to implement security rules. If you have different policy rules, you can create different functions to form a predicate. Depending upon the policy rules defined in the predicate procedure, the resultant predicate is appended to the user's query. The server process now executes this predicated query and sends the results back to the user. Thus the process of query modification is entirely transparent to the user.

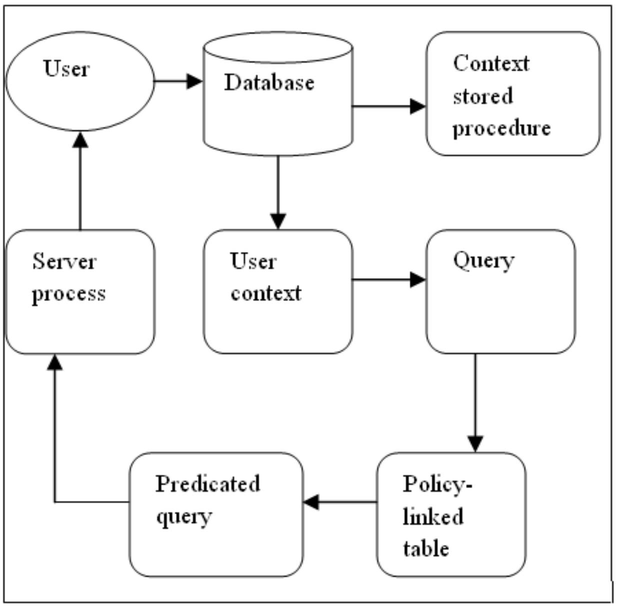

Figure 1: Architecture of Oracle VPD

5. SYSTEM ARCHITECTURE

In this Architecture, we implemented the modules that handle the system during the process of access control at the Database level. The Following Figure 2 shows the System Architecture. Request sent by the user is received by Data source provider which provides a standard data access interface to the user application. User request is intercepted at the data source provider and access control is applied. DS Provider forwards the user request to the Request handler for extraction of access authorizations. Query Interface module presents GUI to build query by selecting data source and applying filters. Request handler fetches the user request from the data source provider. Request from user contains data access request and the set of credentials and sent to Policy Engine for verification and validation. In Policy Store, access control rules are stored. The Policy Store maintains a mapping between the data items and the corresponding credentials required to access those items. Policy Engine module takes user authorization information and query build by user as input and based on policies stored in policy store generates partial WHERE clause of query to be submitted to the data source. Here Access authorization is determined by the Policy Engine. Query Modifier and Generator module combines predicate generated by policy engine to the filter conditions given by user. The modified query is fired to database and result set obtained is according to the authorizations defined. DS Provider sends the data access request to the data source fetches the result from the database and sends it back to the user.

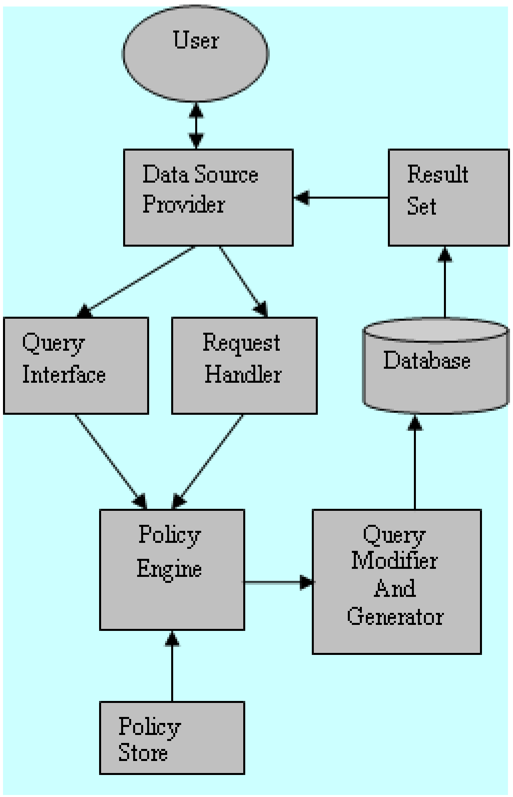

Figure2. System Architecture

6. CONCLUSION

In this paper we applied Different modules exist for providing access control at Database level. We also applied the FGAC model in relational database as a part of DBMS. Any change in policies doesn't affect the application. This paper highlighted the approaches taken to provide fine-grained access control to data in DBMS.

The currently implemented market solution, Oracle's VPD leaves much to be desired. Fine-grained control, on tuple or field level increases management function significantly, and also makes it more complex for these basic schemes. FGAC using Query Modification approach definitely creates complex queries, which surely degrades the performance. Performance evaluation and analysis can also be done on large database in future.

7. REFERENCES

[1] Neha Sehta, Dr. Suresh Jain "A Fine Grained Access Control Model for Relational Databases" IJCSIT, Vol. 3 (1), 2012, 3183 – 3186

[2] Stonebraker, M., Wong, E., 1974. "Access Control in a relational Database Management System by Query Modification" Proc. ACM Annual Conf., p.180-186. [doi:10. 1145/800182.810400]

[3] Surajit Chaudhuri, Tanmoy Dutta, and S. Sudarshan, "Fine Grained Authorization Through Predicated Grants"

[4] Qihua Wang¤ Ting Yu,et al, On the Correctness Criteria of FineGrained Access Control in Relational Databases

[5] E. Bertino, S. Jajodia, and P. Samarati, "A Flexible Authorization Mechanism for Relational Data Management Systems," in ACM Transactions on Information Systems, April, 1999

[6] Nirmal Dagdee, Ruchi Vijaywargiya: "Role based hybrid access control Methodology for shared Electronic Health Records".

[7] Ravi S. Sandhu, Edward J. Coynek, Hal L. Feinsteink and Charles E. Youmank: "Role-Based Access Control Models", IEEE Computer, Volume 29, Number 2, pages 38-47, 1996

[8] Sabrina De Capitanidi Vimercati, Pierangela Samarati: New Directions in Access control, www.spdp.dti.unimi.it/papers/nato.pdf, 2002

[9] Pierangela Samarati1 and Sabrina De Capitani di Vimercati, "Access Control: Policies, Models, and Mechanisms".

[10] Utkarsh Jain, Seminar Report "Fine-grained Access Control in Databases"

[11] The virtual private database in oracle9ir2: An oracle technical white paper. http://otn.oracle.com/deploy/security/oracle9ir2/pdf/vpd9ir2twp.pdf.