

# To Detect Who and When Tamper Data in Database

Chanda Kataria

Scholar (M.Tech, CSE)

Department of Computer Science and Applications  
Kurukshetra University, Kurukshetra

Dr. Kanwal Garg

Supervisor (Assistant Professor)

Department of Computer Science and Applications  
Kurukshetra University, Kurukshetra

**Abstract:** Secure data storage is a requirement of almost every existing organization whether it is public, private or government. Secure data has to deal with many problems for its storage in database such as data consistency, data integrity and many others. Tampering in database is one of the main issue to be solved as soon as possible as it results in corrupted or false data. This paper deals with approach that find tampered data and also find who and when tamper the data. Oracle 10g is used to deal with this problem. The main concern of the paper is to result the name of culprit and the time when data got tampered with.

**Keywords:** Database, Tampering, Oracle, Data storage.

## I. INTRODUCTION

As the use of computer technology has increased in almost every field, the concept of how to store data is also changed from manual to technical. Every term related to data confined itself to one word, i.e. 'Database'. Database is a field designed and developed for data storage in a secure format. Confidential and private data need more security than normal data and database has now developed to such an extent that it can provide protection to every type of data whether private, confidential or normal. The main focus is on to provide measures to stop security violations. Security can be maintained by not letting data to get tampered. Tampering is an activity in which alterations are made, especially secretly so as to harm existing data. By accessing such information which one is unauthorized to access, users try to damage data which leads to data tampering. Log In and password methods has made it possible to don't let any unauthorized user to have access on data, but what if any authorized user say; DBA, tries to harm data who has full rights to access data. In case, if tampering is found in data, we are required to investigate who and when tampered data. To detect whether data is tampered or not, different software are required by the system, one is for notarization and second is for validation. Process flows as, [1] when data is stored in database, it is the duty of notarizer to compute hash value of every tuple and store it in database for further processing. Notarizer perform the same task for every tuple which is inserted in database. Now, to check integrity of data, validator comes to play its role. Validator, after fixed interval of time re-computes hash value for already existing data and matches it with previous obtained hash value of same data. If it does

not match, it means data is tampered. After detecting tampering, it becomes essential to find the main culprit. So for that, investigation approach is followed which provide us with the name of culprit and time when he tampered data. Forensic approach provides many algorithms to find corruption time and name of culprit. Some of the algorithms are:

*A. Monochromatic Algorithm:* In this, only single corrupted region can be found at a time. It uses only single chain which is black chain for every validation event [2] [3].

*B. RGBY Algorithm:* This algorithm uses four more chains with the black chain. Four chains are red, green, blue and yellow. Red and yellow chains are used for every odd order of validation event and blue and green chains are used for every even order of validation event [2] [3].

*C. Tiled-Bitmap Algorithm:* It uses multiple partial chains for every validation which is stored in the form of tiles. It can handle multiple corruption events at a time but it can overestimate the degree of corruption by resulting in false positives [2] [3].

*D. A3D Algorithm:* There is a slowly increase in the number of chains at each validation. Concept of fixed pattern chain is not repeated in this algorithm which helps it to locate only positive corrupted regions while handling multiple corruptions. It overcomes the limitation of tiled-bitmap algorithm. It is an improved version of all forensic analysis algorithms in case of database security [3].

In this paper, oracle 10g is used as a database and concept of triggers is used to find the time when data is corrupted and the name of culprit. Direct triggers are used to solve the problem.

## II. RELATED WORK

Much work has been done in this field. Richard T. Snodgrass, Shilong Stanley Yao and Christian Collberg in 2004 has provided the basic steps of how to detect tampering and in what order notarization and validation event will occur [1]. In 2006, Kyriacos Pavlou, Richard T. Snodgrass explain the basic concept of tamper detection in database by providing tamper detection approach and also

explain basic forensic analysis algorithms to detect tampered locations [2]. KyriacosPavlou, Richard T. Snodgrass in 2008 explain more reliable and accurate forensic algorithms and characterize the 'forensic cost' under worst-case, best-case and average-case assumptions on the distribution of corruption sites of different algorithms [3].ShaguftaRajguru, Deepak Sharma, has explained how to work in oracle environment for detection of tampering [5]. Much more has been done by different researchers in this field and much is about to be done by upcoming researchers.

### III. PROPOSED APPROACH

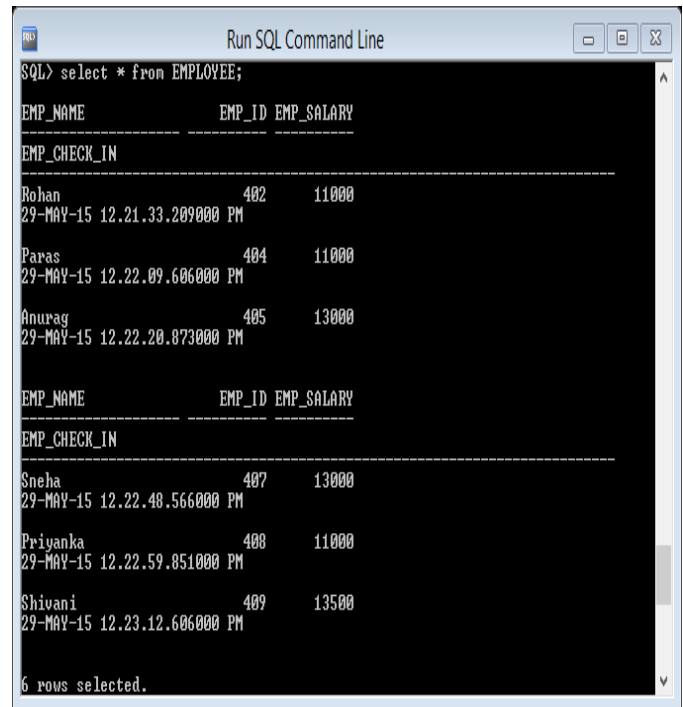
To detect whether data is tampered or not, in Oracle 10g, ORA\_HASH function [6] is used. To find time when tampering is occurred, triggers are used. So, firstly tables are created. Here, one table is created as

```
SQL> CREATE TABLE EMPLOYEE
(emp_namevarchar(20), emp_id number, emp_salary
number, emp_check_in timestamp);
```

This query will create a table named EMPLOYEE. After creating table, insert values in it by simply typing query as

```
SQL> INSERT INTO EMPLOYEE VALUES
('&emp_name', '&emp_id', '&emp_salary',
current_timestamp);
```

The six tuples which are inserted in table are shown in Figure1. We are now required to create backup tables for table EMPLOYEE. In these backup tables, values will automatically get inserted when table is modified or something is deleted from or inserted into it. There is need to apply six triggers,two for insert (before and after), two for delete (before and after) and two for update (before and after).All these triggers will store values in their corresponding backup tables. One backup table is created for insert values (IN\_BACKUP), one is for delete values (DEL\_BACKUP) and two for update values (one is for old values (UPO\_BACKUP) and second is for new values (UPN\_BACKUP)).



EMP_NAME	EMP_ID	EMP_SALARY	EMP_CHECK_IN
Rohan	402	11000	29-MAY-15 12.21.33.209000 PM
Paras	404	11000	29-MAY-15 12.22.09.606000 PM
Anurag	405	13000	29-MAY-15 12.22.20.873000 PM
Sneha	407	13000	29-MAY-15 12.22.48.566000 PM
Priyanka	408	11000	29-MAY-15 12.22.59.851000 PM
Shivani	409	13500	29-MAY-15 12.23.12.606000 PM

Figure1: Table EMPLOYEE

So, four backup tables are required. These are created by typing:

```
SQL> CREATE TABLE IN_BACKUP
(emp_namevarchar(30), emp_id number, emp_salary
number, dat_tm timestamp(6), user_namevarchar(30));
```

```
SQL> CREATE TABLE DEL_BACKUP
(emp_namevarchar(30), emp_id number, emp_salary
number, dat_tm timestamp(6), user_namevarchar(30));
```

```
SQL> CREATE TABLE UPO_BACKUP
(emp_namevarchar(30), emp_id number, emp_salary
number, dat_tm timestamp(6), user_namevarchar(30));
```

```
SQL> CREATE TABLE UPN_BACKUP
(emp_namevarchar(30), emp_id number, emp_salary
number, dat_tm timestamp(6), user_namevarchar(30));
```

Values in these backup tables will automatically get inserted when triggers belonging to these tables will execute. Now triggers are created as

```
SQL> truncate table UPO_BACKUP;
SQL> Create or replace trigger trg_upo
Before update on EMPLOYEE
For each row
Begin
Insert into UPO_BACKUP values
(:old.emp_name, :old.emp_id, :old.emp_salary,
current_timestamp, user);
End;
/
```

The above created trigger (trg\_upo) will store its results in table UPO\_BACKUP. It results in the values that were in table EMPLOYEE before updation. Next trigger (trg\_upn) is created to store new (after update) values of table EMPLOYEE in table UPN\_BACKUP.

```
SQL> truncate table UPN_BACKUP;
SQL> Create or replace trigger trg_upn
After update on EMPLOYEE
For each row
Begin
Insert into UPN_BACKUP values
(:new.emp_name, :new.emp_id, :new.emp_salary,
current_timestamp, user);
End;
/
```

Same triggers are created for insertion and deletion also. Now we need to create users. Suppose, two users are created USERA and USERB and they are granted access to table EMPLOYEE by typing

```
SQL> grant select, insert, update, delete on EMPLOYEE to USERA;
```

```
SQL> grant select, insert, update, delete on EMPLOYEE to USERB;
```

Now when user will connect in, if he will perform any operation on table EMPLOYEE, it will automatically get store in their corresponding operation's backup table. Users have no access to any of the backup table. After that when it is found that someone has tampered the data, the backup tables are checked which will tell about which user has performed what operation on table EMPLOYEE at what time. So with the help of triggers we can know the name of culprit with the timestamp when he/she has tampered the data.

#### IV. RESULTS

Suppose in table EMPLOYEE some changes are found by the validation event. Now it is required to find those changes and who has done that. So DBA or any high authority who has all access to database will disconnect all connected users and proceed its investigation process by accessing those backup tables which were created earlier for triggers. The resulting tables are shown in Figure 2 and in Figure 3.

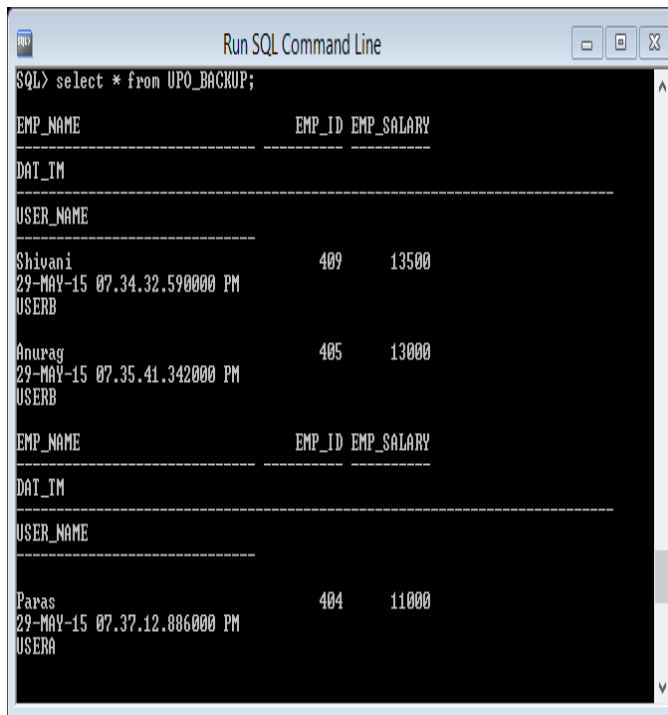


Figure2: Table UPO\_BACKUP

In Figure2, UPO\_BACKUP table is shown which shows all the tuples which are changed by different users. All values in this table are old values. In Figure3, UPN\_BACKUP table is shown which shows all the new values for changed tuples of table EMPLOYEE.

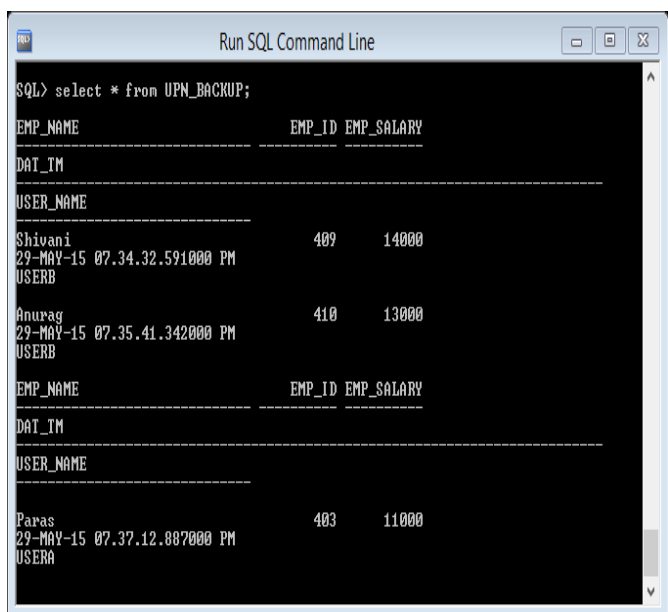


Figure3: Table UPN\_BACKUP

We can compare both tables UPO\_BACKUP and UPN\_BACKUP. In Figure2, emp\_salary for first tuple is changed from 13500 to 14000 by user USERB, emp\_id for second tuple is changed from 405 to 410 by user USERB and emp\_id for third tuple is changed from 404 to 403 by user USERA. Timestamp for all updated tuples is also shown in the tables. So with this we can find which user has performed changes on which tuple and also on which table, if more than one table exists in database. Same type of results can be generated for insert and delete queries also.

## V. CONCLUSION

If security gets violated, there are steps to overcome almost all changes made in any database. One of the methods is explained in this paper. Further, there are many forensic analysis algorithms which are explained in starting of the paper. These algorithms find all the corrupted regions in database. The method explained here in this paper is very easy and basic. It helps many database users to secure their data from unwanted corruptions and interruptions.

## REFERENCES

- (1) Richard T. Snodgrass, Shilong Stanley Yao and Christian Collberg, "Tamper Detection in Audit Logs", Proceedings of the 30<sup>th</sup> VLDB Conference, Toronto, Canada, 2004.
- (2) KyriacosPavlou, Richard T. Snodgrass, "Forensic Analysis of Database Tampering", SIGMOD'06, June 27-29, 2006, Chicago, Illinois, USA. ACM 1-59593-256-9/06/0006.
- (3) KyriacosPavlou, Richard T. Snodgrass 2008, "Forensic Analysis of Database Tampering", ACM Trans. Datab. Syst. 33, 4, Article 30 (November 2008).
- (4) Oracle Database 10g DBA Handbook.
- (5) ShaguftaRajguru, Deepak Sharma, 2014, "DATABASE TAMPER DETECTION AND ANALYSIS", International Journal of Computer Applications (0975-8887), Volume 105 – No. 15, November 2014. URLs
- (6) [http://docs.oracle.com/cd/B12037\\_01/server.101/b10759/functions097.htm](http://docs.oracle.com/cd/B12037_01/server.101/b10759/functions097.htm) [retrieved on 28/5/2015].