iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 2 **Issue:** VI **Month of publication:** June 2014

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

ISSN: 2321-9653

INTERNATIONAL JOURNAL FOR RESEARCH IN APPLIED SCIENCE AND ENGINEERING TECHNOLOGY (IJRASET)

Maintaining Data Security In Privacy Preserving Association Rule Mining

Adesh chaudhary[1] ,Krishna Pratap Rao[1],Prashant Johri [1]

School of Computing Science and Engineering[1],

Galgotias University[1], Greater Noida,India

Abstract - This paper is based on the various data mining techniques which are used to detect some unwanted transaction in data base. Our approach concentrates on mining data dependencies among data items in the database. A data dependency collier is designed for mining data correlations from the database. The transactions not amenable to the data dependencies mined are identified as mischievous transactions. The practical state that the proposed method works effectively for detecting malicious transactions provided certain data dependencies exist in the database. In particular, recent advances in the data mining field have lead to increased agitation about privacy. While the subject of privacy has been uniquely studied with respect of cryptography recent work in data mining field has lead to renewed interest in the field. In this paper, we will introduce the topic of privacy-preserving data mining and provide an overview of the different topics covered in this paper.

KeyWords - Data Mining, Intrusion Detection, Database Security

I. INTRODUCTION

Successful applications of data mining techniques have been demonstrated in many areas that benefit commercial, social and human activities. Along with the success of these techniques, they pose a threat to privacy. One can easily disclose other's sensitive information or knowledge by using these techniques. So, before releasing database, sensitive information or knowledge must be hidden from unauthorized access. To solve privacy problem[1], PPDM has become a hotspot in data mining and database security field. In recent years, data mining has been viewed as a threat to privacy because of the widespread proliferation of electronic data maintained by corporations. It increased trouble about the security of the private or public data. Privacy preserving data mining technique gives novel way to solve this problem.

In recent years, a number of techniques have been proposed for modifying or transforming the data in such a way so as to preserve privacy. The Association rule hiding approach is one of the widely used approach. Association rule hiding problem can be defined as: convert the original database into sanitized database so that data mining techniques will not be able to mine sensitive rules from the database while all non sensitive rules remain visible. It is known that each strong rule extracts from frequent item sets. To prevent sensitive rules (determined by the experts) being mined in the process of association rule mining, many methods are developed, all of which are based on reducing the support and confidence of rules that specify how significant they are. The rest of this paper is organized as follows. We will review the basic concepts of PPDM and different studies performed in the area of PPDM under various categories. We shall concentrate on metrics that are used to measure the side-effects resulted from privacy preserving process in section V. In section VI, we will discuss heuristic based algorithms. Although many different approaches are employed to protect important data in today's networked environment, these methods often fail. One way to make data less vulnerable is to deploy Intrusion Detection System (IDS) in critical computer systems.

ISSN: 2321-9653

In case a computer system is compromised, an early detection is the key for recovering lost or damaged data without much complexity. In recent years, researchers have proposed a variety of approaches for increasing the intrusion detection efficiency and accuracy. But most of these efforts concentrated on detecting intrusions at the network or operating system level. They are not capable of detecting malicious data corruptions, i.e., what particular data in the database are manipulated by which specific malicious database transaction(s). Without this information, fast damage assessment and recovery cannot be achieved.

When an attacker or a malicious user updates the database, the resulting damage can spread very quickly to other parts of the database through valid users. Quick and accurate detection of a cyber attack on a Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the given first page.

Database system is the prerequisite for fast damage assessment and recovery. Our approach concentrates on mining data dependencies among data items in the database. By data dependency we refer to the data access correlations between two or more data items. The techniques employed use data mining approach to create data dependencies among various data sets. dependencies which are generated are in the classification rules form, i.e., before one data item is updated in the database what other data items probably need to be read and after this data item is updated what other data items are most likely to be updated by the same transaction.

The rest of the paper is organized as follows: Section 2 briefly describes the existing research on IDS. We then introduce our classification model in section 3. The mechanism of the proposed data dependency miner is illustrated in section 4. Section 5 presents the formal algorithm for our technique. Section 6 provides some of the results of the experiments.

II. RELATED WORK

An intrusion is defined as the set of actions that are made to maintain the probity, privacy or availability of a resource [9]. Sensitive information or knowledge must be hidden from illegal access this led to the privacy and data security problem, PPDM has become a hotspot in data mining[1]. Some IDS models [2, 3] are representative ones. Artificial intelligence [8] and Data mining [4, 5] applications in intrusion detection are employed by some researchers to reduce the human effort for constructing IDS and to increase the accuracy of the detection. For example, data mining approaches have been employed to detect fraud behaviors in a telecommunication network by Fawcett and Provost [2].Only very limited research has been conducted in the field of database intrusion detection. The Hidden Markov Model has been proposed in [6] to detect malicious data corruption. Lee et al. [5] have used time signatures in discovering database intrusions. Their approach is to tag the time signature to data items. A security alarm is raised when a transaction attempts to write a temporal data object that has already been updated within a certain period. Another method presented by Chung et al. [6] identifies data items frequently referenced together and saves this information for later comparison.3.

III. CLASSIFICATION MODEL

Compared to the existing approach for modeling database behavior [7] and transaction characteristics [6, 8] to detect malicious database transactions, the advantage of our approach is that it's less sensitive to the change of user behaviors and database transactions. It's observed from real -world database applications that although transaction program changes then it is not necessary that whole database structure and data correlations change.

The proposed model is designed to identify malicious transactions submitted to the DBMS by an intruder that bypassed the access control mechanism of a database system. For example, the intruder may get access to a database by employing SQL injection to a poorly coded web-based application or stealing the password of a legitimate user. Thus an intruder can access the database from a remote site by submitting transactions manually or through a different application.

ISSN: 2321-9653

This work is based on the relational database model [6]. A transaction is a logical unit of database processing that includes one or more database access operations. Our model requires that the database log records both read and write operations of each transaction.

IV. THE DATA DEPENDENCY MINER

The data dependency miner performs the analysis of data dependencies among data items in the database. The following definitions help understanding the concept.

A. Data Dependency Terminologies

Because our overall goal is to discover data dependencies that are related to sequence of operations performed by transactions, we first define *sequence* in our context.

Definition 1: A sequence is an ordered list of read and/or writes operations. We denote a sequence *s* by $<o_1(d_1), o_2(d_2), \ldots, o_n(d_n)>$, where $o_i \in \{r, w\}$ and d_k is a data item, $1 \leq k \leq n$. D(s) represents the set of data items contained in the sequence, i.e., $D(s) = \{d_1, d_2, \ldots, d_n\}$. The support for a sequence is defined as the fraction of total transactions that contains this sequence.Read sequence and write sequence are employed to define read and write dependencies respectively.

Definition 2: The *Read Sequence* of data item *x* is the sequence with the format $<r(d_1), r(d_2), \ldots, r(d_n), w(x)>$ which represents that the transaction may need to read all data items $d_1, d_2, \ldots, d_n$ in this order *before* the transaction updates data item *x*. It must be noted that each data item may have several read sequences each having different length. All these sequences together are called *Read Sequence Set* of this data item.

The notation rs(x) is used to denote the read sequence set of data item *x*. For example, consider the following update statement in a transaction.

Update Table1 set x = a + b + c where d = 90;

In this statement, before updating *x*, values of *a*, *b*, *c* and *d* must be read and then the new value of *x* is calculated. So <r(a), r(b), r(c), r(d), w(x)> rs(x).

It must be noted that the database log only contains before and after images of *x* instead of the mathematical operation used for calculating *x*, i.e., x = a + b + c. The above example is only for illustrating the concept of read sequence. The database log containing the above transaction may actually look like:

T1: r(m), r(n), w(y), r(u), r(v), r(a), r(b), r(c), r(d) w(x), r(a), w(c), commit.

Before the write operation w(x), 8 data items have been read. Some of them may not have data dependencies with *x*, e.g., they are read by another SQL statement in the same transaction. This means the new value of *x* is not directly dependent on the values of all these 8 data items. Our goal is to determine that in order to update *x*, data items *a*, *b*, *c*, and *d* are most likely need to be read and are relevant for calculating the new value of *x*. It must be noted that the mining result may only illustrate *a* and *b* have data dependencies with *x*. This may happen when some other transaction only read values of *a* and *b* before updating *x*.

Definition 3: The *Write Sequence* of data item *x* is the sequence with the format $< w(x), w(d_1), w(d_2), \ldots, w(d_n) >$ which represents that the transaction may need to write all data sets $d_1, d_2, \ldots, d_n$ in this order *after* the transaction updates data item *x*. It must be noted that each data item may have several write sequences each having different length. All these sequences together are called *Write Sequence Set* of this data item.

For example, consider the following update statements in one transaction.

Update Table1 set x = a + b + c where …

Update Table1 set y = x + u where …

Update Table1 set z = x + w + v where …

Using the above example, it can be noted that <w(x), w(y),w(z)> is one write sequence of data item *x*, that is <w(x), w(y),w(z)> ws(x), where ws(x) denotes the write sequence set of *x*.

Definition 4: The *Weight of Data Dependency* indicates to what extent a data item *x* depends on other data items. It's defined by the possibilities of reading (writing) these data items before

ISSN: 2321-9653

(after) updating *x*. The notions *rweight*(x, D(s) - x) and *w weight*(x, D(s) - x) denote the weight of read dependency and write dependency respectively. A pre-set threshold is used to identify whether a dependency is weak or strong.

For example, suppose the predefined threshold for weight of data dependency is 40%. For the sequence <r(a), r(b), r(c), r(d), w(*x*)>, if the probability of reading {a, b, c, d} before *x* is updated is 75%, then *r weight*(x , {a, b, c, d}) is equal to 75%. Since this is larger than the threshold, we say, the dependency between *x* and {a, b, c, d} is strong.

Figure 1 illustrates an example data dependency. Data item *x* has read dependency relationships with {a, b, c, d}, {c, d}, and {x, e, f}. Besides, it has write dependency relationships with {y, z} and {u, v}. Suppose the predefined threshold for weight of data dependency is 40%. Then for the read dependency only {a, b, c, d} has strong data dependency with *x*. Similarly for the write dependency only {u, v} has strong data dependency with *x*.

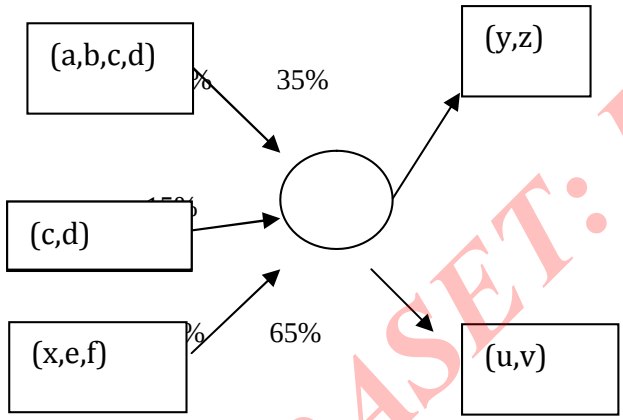


Figure 1. Example Data Dependency

V. ALGORITHM

Initialize the read sequence set RS ={} and the write sequence set WS= {}

Initialize the read rule set RR={} and the write rule set WR={}

Generate the sequential patterns X = {x_i | support(x_i) > minimum support} by using existing sequential pattern mining algorithm

for each sequential pattern x_i where | x_i | > 1

if there's a write operation in it for each write operation $w_i \in x_i$

if <r(d_{i1}), r(d_{i2}), r(d_{i3}),…, r(d_{in}), w(d_i)>$\notin$ RS and

< r(d_{i1}), r(d_{i2}), r(d_{i3}),…, r(d_{in})> $\neq$ <$\emptyset$>

add <r(d_{i1}), r(d_{i2}), r(d_{i3}),…, r(d_{in}), w(d_i)> to RS where r(d_{i1}), r(d_{i2}), r(d_{i3}),…, r(d_{in}) are all read operations before w(d_i)

if <w(d_i), w(d_{j1}), w(d_{j2}), w(d_{j3}),…, w(d_{jk})> $\notin$ WS

and <w_{j1}, w_{j2}, w_{j3},…, w_{jk}> $\neq$ <$\emptyset$>

add <w(d_i), w(d_{j1}), w(d_{j2}), w(d_{j3}),…, w(d_{jk})> to WS where w(d_{j1}), w(d_{j2}), w(d_{j3}),…, w(d_{jk}) are all write operations after w_i

for each sequence in RS

if support(<r(d_{i1}), r(d_{i2}), r(d_{i3}),…, r(d_{in}), w(d_i)>) /support(<w_i(d_i)>) > minimum confidence

add w(d_i)→r(d_{i1}), r(d_{i2}), …, r(d_{in}) to RR for each sequence in WS

if support(<w(d_i), w(d_{j1}), w(d_{j2}), …, w(d_{jk})>) /support(<w_i(d_i)>) > minimum confidence

add w(d_i)→w(d_{j1}), w(d_{j2}), …, w(d_{jk}) to WR

Steps 1 and 2 initialize the read/write sequence sets and read/write rule sets respectively. Step 3 employs existing sequential pattern mining algorithm to generate sequential patterns consisting of a sequence of read and/or write operations that satisfy the minimum support. Step 4 generates read and write sequence sets from the sequential patterns mined. The sequential patterns that don't contain any write operation are not considered during this phase. For each write operation in all other patterns, the sequence consisting of all read operations before this write operations and this write operation itself are added to the read sequence set. Similarly the sequence

consisting the write operation itself and all write operations after this write operation are added to the write sequence set. Step 5 generates read and write rules from the read and write sequence sets based on the minimum confidence. All the rules satisfying minimum confidence are found in the read rule or write rule sets as the output of the algorithm.

VI. EXPERIMENTAL ANALYSIS

Several experiments were conducted for testing the performance of the proposed method. Two different database logs were generated for testing both true positive rate and false positive rate of our model. The first log consisted of synthetic database transactions, which were known as bitter transactions. These malicious transactions were generated randomly based on the assumption that the attacker might not have the knowledge of data dependencies in user database. The second log consisted of normal user transactions.

Table 5 illustrates the baseline setting of our experiment. In order to test the extent to which our approach is sensitive to data dependencies, the malicious and normal user transactions were generated based on the third and the fourth parameters in Table 1, i.e., the average number of read operations immediately before the write operation in transactions and the average number of write operations in transactions. Then, by varying one parameter at a time, we evaluated how the intrusion detection system responded to the change of the parameter and whether the performance was sensitive to this change.

Support	confidence	read opr^n	write opr^n	transaction
0.10	0.70	1.5	1.5	1900

Table 1. Baseline Setting of Experiments

Figure 1 presents the true positive rates in detecting malicious transactions. The database log containing malicious transactions was used for this experiment. Figure 1 shows that the true positive rate increases steadily when the data dependency is stronger among data items. It is noted that when the average number of write operations varies from 1 to 5, the true positive rate increases from 36% to 86%. Whereas, when the average number of read operations immediately before a write operation in transactions varies from 1 to 5, the true positive rate increases from 60% to 81%. Comparing to the two graphs in Figure 1, it is observed that the true positive detection rate is more sensitive to the average number of write operations in a transaction. That is, if there are more update statements in transactions, the detection rate will increase quickly.

Figure 3 illustrates the false positive rates in testing normal user transactions. The database log containing normal user transactions was used in these cases. It is observed that the false positive rate can be as low as 12% when the data dependency is not very strong. The maximum false positive rate is 25% when the average number of read operations immediately before a write operation in transactions is 5. Comparing to Figure 2, it is observed that with the increase of the data dependency, the increase of the true positive rate is much higher than that of the false positive rate.

VII. CONCLUSION

In this paper we proposed a data mining approach for detecting malicious transactions in database systems. Our approach concentrates on mining data dependencies among data items in the database. Data dependency rules discovered by the data dependency miner are employed as classification rules for identifying anomalies. The experiment on synthetic database transactions illustrates that the proposed method works effectively for detecting malicious transactions in database systems provided certain data dependencies exists. The result further shows that the stronger the data dependency among data items, the better the overall performance.

REFERENCES

CHIRAG N. MODI, UDAI PRATAP RAO, MAINTAINING PRIVACY AND DATA QUALITY IN PRIVACY PRESERVING ASSOCIATION RULE MINING. 2010 SECOND INTERNATIONAL CONFERENCE ON COMPUTING, COMMUNICATION AND NETWORKING TECHNOLOGIES.

[2]. Forrest, S., Hofmeyr, S. A., Somayaji, A., and Longstaff, T. A. A Sense of Self for Unix Processes. In Proceedings of the 1996 IEEE Symposium on Security and Privacy, pages 120–128, IEEE Computer Society Press, 1996

[3]. Javitz, H. S. and Valdes, A. The SRI IDES Statistical Anomaly Detector. In Proceedings of the IEEE Symposium on Security and Privacy, May 1991

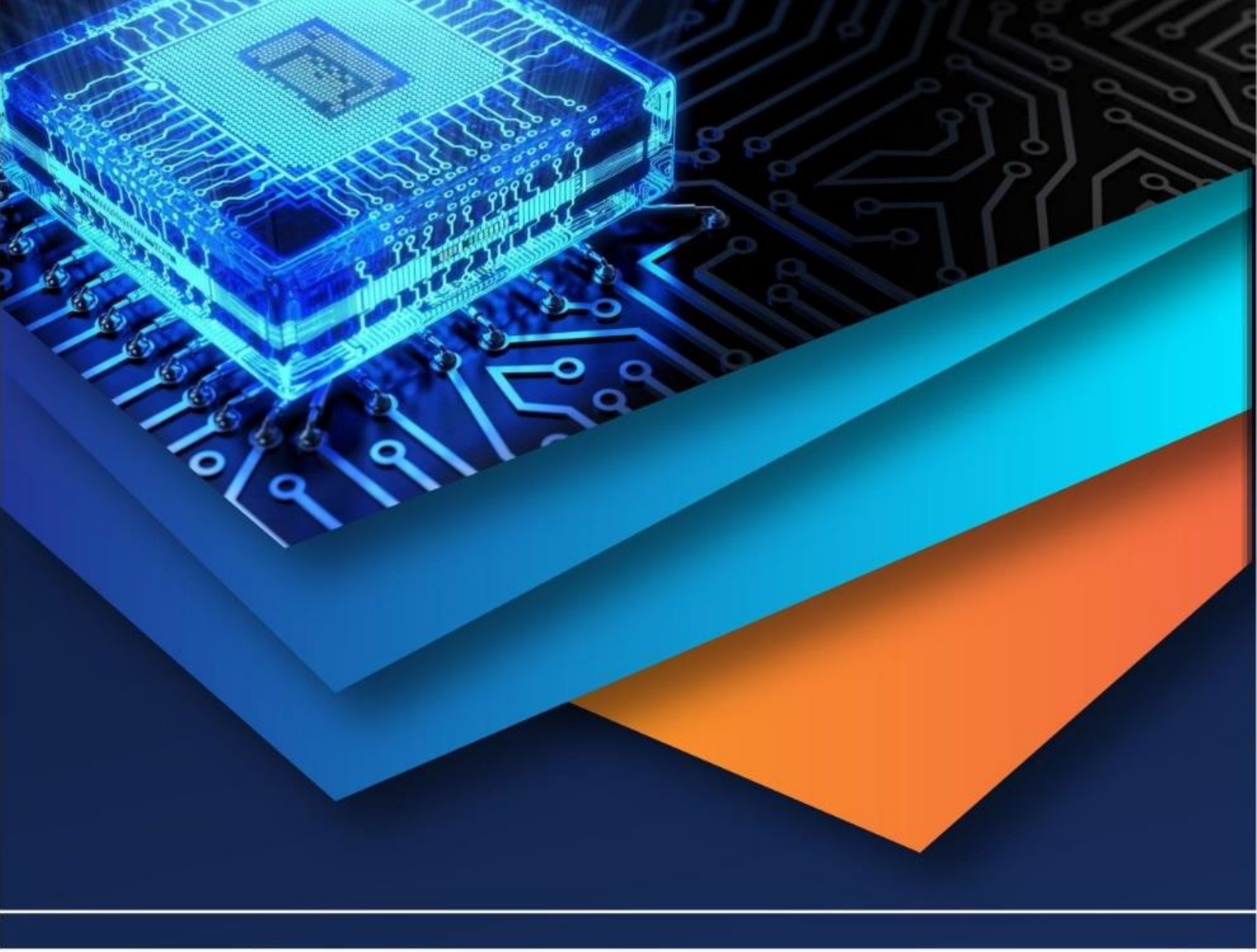
[4]. Lee, W., Nimbalkar, R.A., Yee, K.K., Patil, S.B., Desai, P.H., Tran, T.T., and Stolfo, S.J. A Data Mining and CIDF Based Approach for Detecting Novel and Distributed Intrusions. In Proceedings of 3rd International Workshop on the Recent Advances in Intrusion Detection, October 2000

.[5]. Lee, W. and Stolfo S. Data Mining Approaches for Intrusion Detection. In USENIX Security Symposium, 1998.

[6]. Chung, C., Gertz M., and Levitt, K. DEMIDS: A Misuse Detection System for Database Systems. In Third Annual IFIP TC-11 WG 11.5 Working Conference on Integrity and Internal Control in Information Systems, Kluwer Academic Publishers, pages 159-178, November 1999. Codd, E. A Relational Model for Large Shared Data Banks. In Communications of ACM, June 1970.

[7]. Barbara, D., Goel, R., and Jajodia, S. Mining Malicious Data Corruption with Hidden Markov Models. In Proceedings of the 16th Annual IFIP WG 11.3 Working Conference on Data and Application Security, Cambridge, England, July 2002.

[8]. Lee, V. C.S., Stankovic, J. A., Son, S. H. Intrusion Detection in Real-time Database Systems Via Time Signatures. In Proceedings of the Sixth IEEE Real Time TechnologyApplications



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY