User-independent and Self-optimizing Intrusion Detection Framework for Large Database Systems

REZA ADINEHNIA, NUR IZURA UDZIR, LILLY SURIANI AFFENDEY, ISKANDAR ISHAK, ZURINA MOHD HANAPI
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
Serdang, 43400 Selangor
MALAYSIA
reza.adinehnia@gmail.com, {izura, lilly, iskandar_i, zurinamh}@upm.edu.my

Abstract: - Despite various access control approaches, databases are still vulnerable to intruders who are able to bypass these protective methods and access data, or prevent insiders like authorized users who misuse their privilege. To prevent all such intrusions, this study proposes a multilayer profiling method to provide suitable and reliable valid patterns to be used in the proposed database intrusion detection framework. With the help of association rule learning and Naive Bayes classifier this framework can provide a considerable rate of intrusion detection. The main contributions of this paper are summarized in a granular profiling structure and a detection framework that helps to detect database intrusions even if they are initiated by insiders.

Key-Words: - database intrusion detection, query profiling, data mining, apriori

1 Introduction

Database systems are widely utilized to systematically store information nowadays. The popularity is for the methods database systems use to organize data in terms of storage and retrieval. When it comes to organizations' privacy issues, having security mechanisms in place is a must. There are several databases that need to be accessed globally nowadays. Using mobile communications makes it easier to access information from anywhere, and unfortunately, this advantage is available to hackers and intruders as well. Although several security techniques like access control can prevent unauthorized accesses but they are not efficient against novel attacks [1] especially if they are originated from insiders who are considered as legitimate users. They are normally internal users of the company who have the right access permissions to the data and are aware of existing security mechanisms and policies of the organization. In such a situation none of the encryption and access control techniques can stop them from misusing their privileges.

In a crowded multi-user environment, multiple requests are generated for the same database. When multiple requests come to the database it gives a chance to attackers to identify the inter-transactional deviation. It should not be forgotten that the individual requests and the normal user requests are similar. Thus, a considerable part of database intrusion detection is inter-transactional and if it is combined with the rest of normal database transactions it helps us to produce a more efficient intrusion detection system. As mentioned above, focusing on a single feature like data dependencies or the structure of requests alone is not effective when it comes to insider threats.

One of the most considered methods for detecting database attacks is behavioral analysis. Therefore, specific profiles for the critical assets of the database need to be generated. This process can be automated using data mining methods. Profiles can represent characteristics of a user or a group of users. However, the features can vary for different organizations or departments. That is the reason intrusion detection systems need to be adaptive to the changes and various requirements of the organizations.

Profiling behavioral activities is achieved through recording user access to different objects or different levels of each object in a database. More detailed monitoring produces more accurate profiles. For example, a profile of tables accessed by user is normally less accurate compared to the one with accessed attributes of the database tables. On the other hand, queries that come in the form of a transaction also have their own effects or changes on database. Therefore, a transaction may have a series of modifications that follow a specific purpose. With the fact that different roles, representing a group of users or a department of the organization, may have similar behavior dealing with database through their transactions, it is important to analyze the structure of transactions and add it to the profile features.

A combination of two query profiling method is introduced in this study to form accurate profiles at transaction level. These profiles are used for further tasks to produce a comprehensive intrusion detection framework for database systems.

The rest of this work is organized in four sections. Related works section reviews the related works and discusses the gaps and limitations related to database intrusion detection systems. Methodology describes all the involved procedures from data collection to the applied algorithms. It covers dataset preparation, proposed query profiling method, and data mining approach which is used for this experiment. Implementation and results are then discussed. Finally, Conclusion section provides a summary of the presented solution and highlights the potential future directions.

2 Related Works

Intrusion detection systems were first introduced in the early 1980s and became an active research area in the large number of researches afterwards. Several recent research works have applied data mining for the purpose of intrusion detection. One of the very first works that uses data mining for intrusion detection is introduced in [2], which mainly focuses on link and sequential analysis. Barbara et al. [3] have developed a testbed to detect network intrusions in which data mining is the key technique. Malicious data corruption is determined by using Hidden Markov Model (HMM) and time series in [4]. The proposed model is a database behavioral model that uses HMM to denote the changing behavior over time.

Nevertheless, very few of these researches focused on intrusion detection at the database level. An intrusion detection system called DEMIDS is introduced in [5] which uses working scopes to differentiate queries. They calculate a distance that shows how close a group of attributes is to their related working scopes. However, results do not cover the granularity level of the features in the itemsets.

Lee et al. [6] proposed a real-time database intrusion detection model in which the update time of temporal objects is monitored. An alarm is triggered in case of any suspicious period of update. A two-layer detection technique [7] is also proposed in which authors use behavior modeling at both Web server and database levels. Although a very robust model is proposed, the granularity levels are not tested in their work.

In [8], a fingerprint of legitimate database accesses is used to detect malicious requests. Further queries are considered as attacks if they do not match the existing patterns. The accuracy of the fingerprints is a challenge in this work as it is difficult to provide accurate patterns. Data exploitation is detected using Hidden Markov Model by the authors in [9]. In their system a database behavioral model is created to form attack signatures.

A new misuseability weight measure called M-score is proposed by Harel et al. [10] to predict the risk coming from insiders. This measure defines a weight for the tabular data.

Constraint and dependency graph as well as threat prediction graph are introduced in [11] and used to address the path to unauthorized information that is normally used by insiders. Their proposed mechanism can prevent threats up to 30%. However it is limited to the constraints and dependencies of data items.

Query templates are used for user profiling in [12]. Profiles are created at transaction level by a simple structured template consisting of four records for query type, attributes, tables, and constraints. However, the template is not scaled to cover different levels of access patterns in detail.

A database intrusion detection system is proposed by Bertino et al. [13] in which different levels of profiling is used. Each profile level uses a set of features like query type, number of tables, table names or number of accesses. These profiles are used for anomaly detection. However, their focus is limited to query level. Transaction-level attribute dependency [14] is used in order to detect intrusions. New transactions are detected as intrusions if they do not match the dependency rules. Srivastava et al. [15] improved the concept of attribute dependency. They proposed a weighted data mining algorithm to find dependencies among sensitive attributes. However, using weighted mining method may miss those attributes that are rarely being accessed on database.

Profile granularity is highly dependent on the organization and its users. Profiles in organizations with a large number of users are normally created at role level, but with small number of users, it can be kept at user level, which is more accurate. A role level profiling is used in [13]. However, an insider can bypass the detection system if the profile is kept at role level.

To enhance the capability of database intrusion detection and the problem of insider attacks highlighted above, a multi-layer profiling method is introduced in this study covering both context and result-set based approaches as well as query structure in terms of profile granularity. The benefit of using this model is that it uses the features which help to increase the accuracy and consequently reducing false positive rate. The proposed profiling happens at query and then transaction level. This helps to avoid dependencies on user roles and make the system effective against insider threats.

3 Methodology

In this section all actions required to implement the solution and achieve results are explained. The first step is to prepare the dataset from database log file which is explained in the following section. Then data mining is applied to generate the profiles. The overall training process is shown in Fig.1.

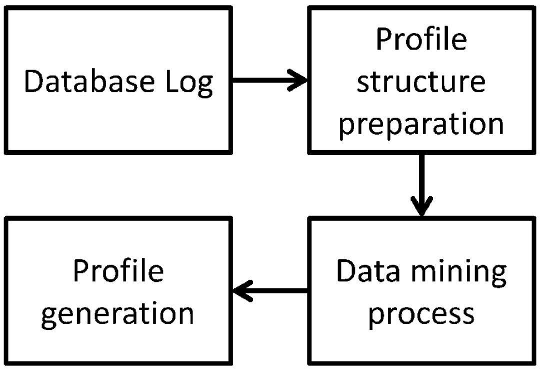

Fig. 1: Training process

3.1 Dataset

A customized database is created in MS SQL Server and filled with about eight million records. Several queries are then generated in the form of transactions in order to prepare the database log. This log file is used as the main dataset for the purpose of data mining. The precise number of queries is 900, with which 500 transactions are generated and launched randomly on the database. These queries are mined to produce the valid or normal patterns and we consider the database log as an intrusion-free dataset. The reason transaction is selected as the focused object of our study is that in some cases a single query may not be considered as anomaly but the overall result of a transaction that consists of several queries can be a part of an abnormal activity and should be considered as intrusion. Transactions contain a number of queries that help to have enough data relationships to produce an acceptable range of itemsets for data mining process. One hundred malicious transactions are also generated to check the accuracy of detection. This is achieved by comparing normal and abnormal patterns using a classifier. The malicious transactions include queries in which abnormal access patterns can be found. For example a select *, delete * or drop table command cannot be found in frequent itemsets and should be considered as intrusion.

3.2 Transaction profiling

Although the database log file contains all activities done by running queries, a proper extraction and interpretation process is needed in order to prepare a suitable query structure to build the valid patterns. Profiling has been introduced and implemented in several works recently which is reviewed earlier in this paper. Profiling is used when the detection method is based on the behavior of user, group of users (role) and queries itself. Therefore, the main issue that arises here is to select suitable features of the log file to increase the accuracy level of detection and reduce errors like false positive rates. Profiling can be performed at the query level, context level or even result-set level. Two different structures are selected in this study in order to prepare the profiles. One is a combination of query-based and context-based profiling. And the other is a combination of query-based and result-set based method. These two methods are then compared in terms of detection accuracy, and then combined to check whether it can produce better detection rate. The one with higher detection rate is used to form the final intrusion detection framework.

From query structure these fields are selected: SQL command, attributes, table name and attributes in where clause. The timestamp of the query is also selected as a context-based item. The overall output is a line for each query in a transaction: Ti (SQL_CMDj, ATTRj, TBLj, ATTR2j, TIME), where Ti is a transaction that may consist of one or more queries. For the result-set based method the number of updated rows (RS) is selected as a feature to determine a normal pattern based on SQL command, table name and its result-set. It helps us to find a relation between each command-table set and the number of rows affected by them. For example, if no row is affected on a table for which we have always more than zero updated rows in normal access patterns, it can be concluded as an intrusion. The structure is like Ti (SQL_CMDj, TBLj, RSj). The first profile structure is named PF1 and the second structure PF2 for ease of referencing to the profile in the following sections.

3.3 Mining process

In this section the overall process of associate rule mining for generating valid patterns is explained. The effectiveness of rule mining is discussed in [16] with an approach to form a better representation of complex redundancy relationships, called PARAS. Moreover, to find a suitable association mining method an experiment is performed by Adinehnia et al. [17] shows that apriori algorithm produces patterns with good level of accuracy. Results in [18, 19] also approve the advantage of using apriori for dIDS. The first stage in the model proposed in this paper is to mine dataset with apriori algorithm and produce patterns to be used later as valid signatures.

For better understanding of the mining procedure, the apriori algorithm concept and mechanism should be explained here. As Fig.2 shows, apriori algorithm process for mining datasets starts with initial frequent item phase in which initial item sets enter the system. Then the first candidate set is generated in the second phase and is filtered in candidate pruning phase [20]. The output of support calculation phase is sent to the candidate generation and the process continues until the desired candidate set is defined.

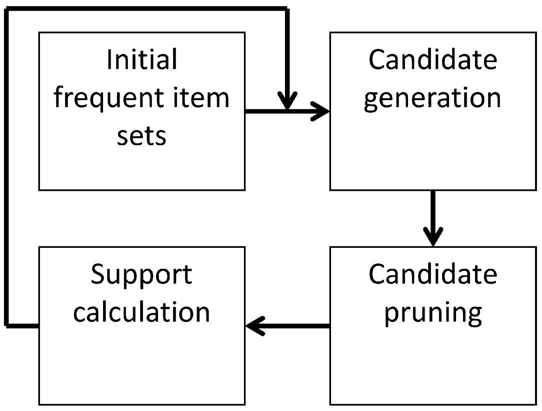

Fig. 2: Process flow of the data mining system

Candidate generation is the process of building new generations from candidates of the previous generation. Each new candidate can be developed into a new potential frequent item set, because it is built from the previous generation candidates that now have more confidence to be the target frequent itemsets. This is stated formally in [20] and represented as follows:

```
$\forall\ f_1, f_2 \in F_k\ do$
    $with\ f_1 = (i_1, \ldots, i_{k-1}, i_k)$
    $and\ f_2 = (i_1, \ldots, i_{k-1}, i_k^*)$
    $and\ i_k < i_k^*$
        $f := f_1 \cup f_2 = (i_1, \ldots, i_{k-1}, i_k, i_k^*)$
```

in which f is a new generation built from candidates of the previous generation. Those candidates that have a difference in their final element are selected to produce the next candidate generation [20]. In the next step of candidate generation, generating each new candidate from two candidates in the previous generation is guaranteed. It is shown below that every subset of each candidate exists in the previous generation:

$$\forall i \in f : f - \{i\} \in F_k$$

The third phase in Fig.2 is the support calculation for the algorithm in which the whole database is examined to find the number of occurrences for each candidate by comparing them with each database transaction [20]. There will be an increment for candidate’s support count if the itemset is found in the transaction:

```
$\forall t \in T\ do$
    $\forall c \in C\ do$
        $if\ c \in t$
            $support\ (c)++$
```

Data complexity is the main problem for the apriori algorithm [17, 20]. The comparison process is very complex and time consuming. However, several attempts [19, 21] have been made to overcome this problem, lead to an enhanced version of apriori algorithm which is proposed in [21]. Apriori is used as the main mining method in this study for generating access patterns to be used in the proposed database intrusion detection framework.

4 Experiment and Results

Both query profiling techniques are implemented using the same dataset and mining process as explained above. Each structure for query profiles is built for the extracted rules from the database log. These rules are the frequent itemsets generated from database using apriori algorithm. The profile structures then form the valid and normal patterns. Java programing language is used for this purpose. The same profiling process is performed on incoming transactions in order to check whether they are intrusions or not. If the pattern generated from the new transaction is found in the stored list of profiles (patterns) then it is considered as a normal transaction. But if it does not match any profile inside the profile repository, it is suspected to be an intrusion. Naive Bayes classifier is used in this experiment at this stage, to determine the probability of being intrusion. This classifier helps us to decide whether the new pattern is an intrusion by providing the amount of similarity between the new pattern and the existing ones. Accuracy and effectiveness of Naive Bayes classifier are proven in [22, 23]. The process of implementing the proposed solution is shown in Fig.3.

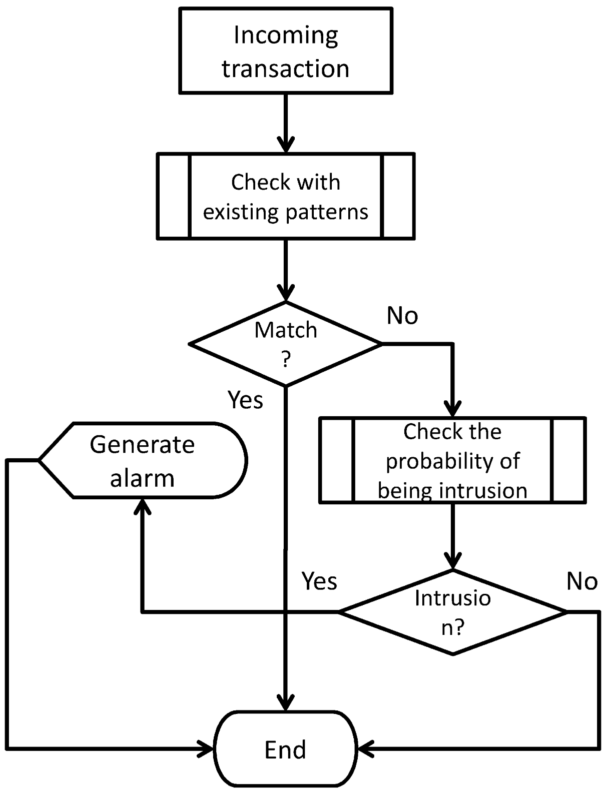
Incoming transaction
Check with existing patterns
Match ?
No
Yes
Generate alarm
Check the probability of being intrusion
Yes
Intrusion?
No
End

Fig. 3: Intrusion detection flowchart

The proposed database intrusion detection system is implemented for each profile structure separately and a third experiment is done for a combination of both PF1 and PF2. A fixed number of transactions are used for all three experiments. Effectiveness of these structures is examined and discussed in the next section by applying a variable amount of transactions. The results for the mentioned profiles are shown in Table 1, in which the highest successful detection rate is 69%.

An interesting observation of this experiment is that the lowest false positive rate is 1% which is observed in PF1. This profiling structure consists of query structure based and context-based profiling methods. Equation 1 is used to calculate false positive rate.

$$False\ positive\ rate = \frac{FP}{FP+TN} \times 100 \qquad (1)$$

where FP and TN are the number of false positive and true negative respectively.

Similarly true positive rate is calculated through the following formula in Equation 2.

$$True\ positive\ rate = \frac{TP}{TP+FN} \times 100 \qquad (2)$$

As it is observed result-set-based approach produces a high false positive rate even if it is used with a query-structure-based method. The FP rate for PF2 is 7%. However, a combination of PF1 and PF2 provides the best detection rate, regardless of the FP rate which is slightly higher than PF1.

Table 1: Detection rates for each profiling method

Profiling method	PF_1	PF_2	PF_1+PF_2
# of transactions	500	500	500
# of malicious transactions	100	100	100
# of intrusions detected	63	77	79
# of intrusions detected correctly (TP)	58	42	69

In the second part of our experiment it is decided to inject variable numbers of transactions to the proposed dIDS in order to determine the effectiveness of each profile structure when the amount of training transactions is changing in the dataset. It is observed that generally the false positive rate is reduced when more samples are used in the training phase. But the question is how much reduction can be achieved for each profiling method. In order to find out the answer the experiment is repeated five times for each profile structure with transactions ranging from 100 to 500. Results of this experiment are shown in Fig.4. It is observed that more accurate training can be achieved using PF1 with a lower number of transactions. However, as the number of transactions exceeds 300 a good level of reduction in FP is observed when a combination of PF1 and PF2 is used. Also, this profile results in a higher detection rate that is represented in Fig.5. Therefore, this combined structure is selected for the final framework of the proposed database intrusion detection system. Results in Fig.4 and 5 are scaled up for better representation.

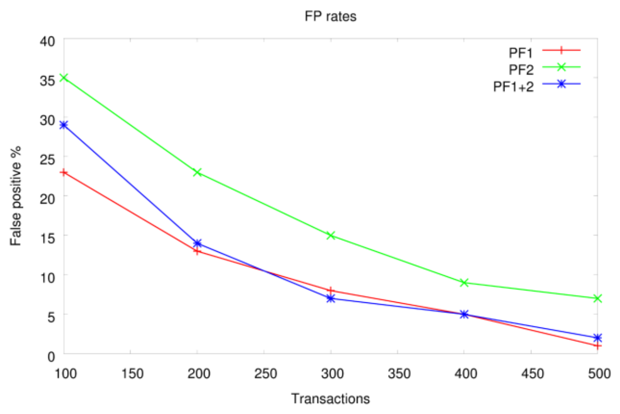

Fig. 4: False positive rates vs. number of transactions

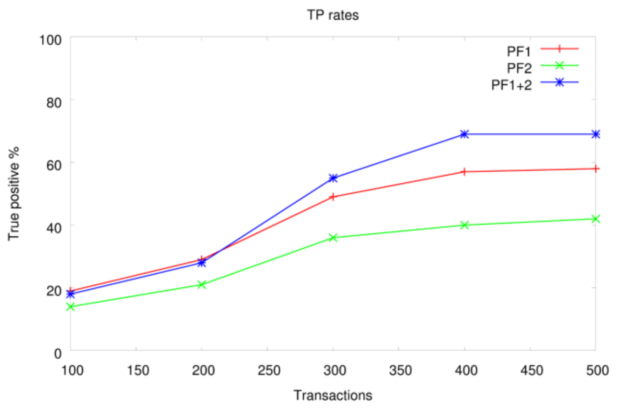

Fig. 5: True positive rates vs. number of transactions

A new framework is then implemented using the final version of the profile introduced above. An intrusion repository is also applied in the proposed framework in which a copy of all newly-detected intrusions is stored. That means whenever a new transaction is detected as an intrusion a new record of its pattern is added to the repository. As it is shown in the proposed framework (Fig.6), incoming transactions are compared with these patterns first and will be passed to the next level only if the system cannot find a match for them in the intrusion repository. Therefore, no further action is required to examine the transaction. This repository not only reduces the time of detection process for repeated intrusions, but considerably increases the detection rate too, see Fig.7.

To implement the work the same dataset described earlier is used but the transactions are injected to the system randomly this time and for a fixed period of time. That means all valid and malicious transactions are entered and repeated several times in the dIDS. Results of this experiment are represented in Fig.7, which shows that all intrusions are detected after about 30 minutes of processing. The number of transactions increased from 720 to 36000 within 50 minutes. Although this result is achieved in a controlled environment with a fixed number of intrusions, it can still be considered as a proof for accuracy of the proposed detection framework.

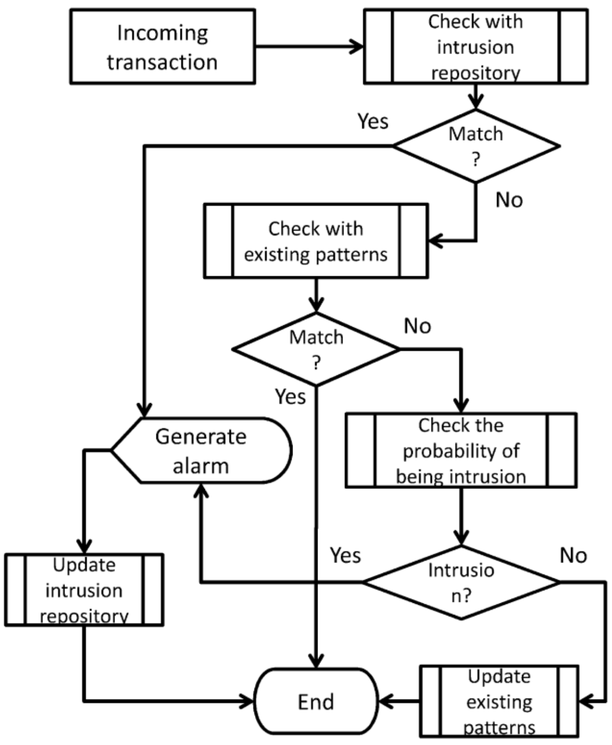
Incoming transaction
Check with intrusion repository
Match ?
Yes
No
Check with existing patterns
Match ?
No
Yes
Generate alarm
Check the probability of being intrusion
Update intrusion repository
Yes
Intrusion?
No
End
Update existing patterns

Fig. 6: Framework of the proposed dIDS

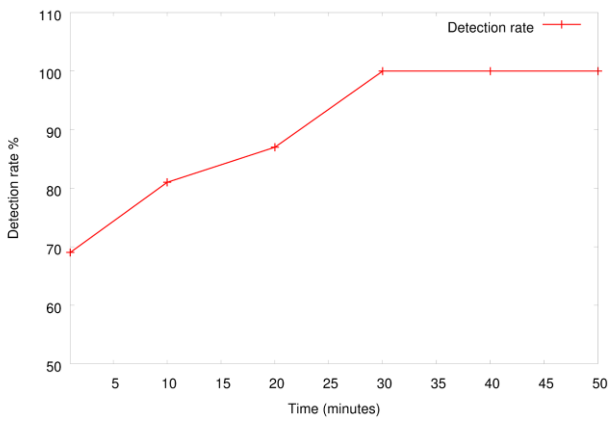

Fig. 7: Detection rate while using intrusion repository

5 Conclusion

Intrusion detection at database level is becoming the main focus point in both organizations and research works. It is because of the importance of protecting sensitive data from malicious accesses. It becomes more important when the source of intrusion is from insiders. Insider attackers have all required permissions to access and modify data legally. So it is almost impossible to detect or prevent them with normal access control methods.

In this study, a comprehensive detection framework is proposed using a multi-layer profiling technique. A proper combination of query-structure-context-based and query-structure-result-set-based approach helps to increase the accuracy of intrusion detection. The valid patterns generated from these profile structures are then compared with new incoming patterns. If they do not match they are sent to the Naïve Bayes classifier to check the probability of being intruded. If any of these transactions are detected as intrusions an alarm will be triggered for administrator to take further appropriate action. The overall results obtained from the proposed technique and framework show an interesting reduction in false positive rate and total successful detection rate of 69% for the dataset introduced in this study. Furthermore, it is observed that the system is able to detect all predefined intrusions in a short period of time after it starts processing the transactions.

The profiling technique proposed in this work can be extended into applying more features of the query structure and other result-set items like the value of the data. The framework can be also optimized in terms of performance. This is vital when the system is dealing with very large datasets. There is also a chance to increase the detection rate if the system can be combined and customized with an organization's policy repository. All these enhancements are considered as future work for this study and are the current research focus of the authors.

References:

[1] Xiaowei Li, Xujie Si, and Yuan Xue. Automated black-box detection of access control vulnerabilities in web applications. *In Proceedings of the 4th ACM conference on Data and application security and privacy (CODASPY '14)*. ACM, New York, NY, USA, 49-60. DOI=10.1145/2557547.2557552, (2014)

[2] Lee W., Stolfo S.J., Data Mining Approaches for Intrusion Detection, *Proceedings of the USENIX Security Symposium*, (1998) pp. 79-94.

[3] D. Barbara, J. Couto, S. Jajodia, N. Wu, ADAM: A Testbed for Exploring the Use of Data Mining in Intrusion Detection, *ACM SIGMOD*, (2001) pp. 15-24.

[4] D. Barbara, R. Goel, S. Jajodia, Mining Malicious Data Corruption with Hidden Markov Models, *IFIP WG 11.3 Working Conference on Data and Application Security*, (2002) pp.175-189.

[5] Chung, C.Y., Gertz, M., Levitt, K., DEMIDS: A Misuse Detection System for Database Systems. *In Proceedings of the integrity and internal control in information system*, (1999) pp. 159–178.

[6] Lee, V., Stankovic, J., Son, S., Intrusion Detection in Realtime Databases Via Time Signatures. *In Proceedings of the 6th IEEE real-time technology and applications symposium (RTAS)*, (2000) pp. 124–133.

[7] Wenhui, S., Tan, T., A Novel Intrusion Detection System Model for Securing Web-Based Database Systems. *In Proceedings of the 25th annual international computer software and applications conference (COMPSAC)*, (2001) pp. 249–254.

[8] Lee, S.Y., Low,W.L.,Wong, P.Y., Learning Fingerprints for Database Intrusion Detection System. *In Proceedings of the 7th European symposium on research in computer security*, (2002) pp. 264–280.

[9] Barbara, D., Goel, R., 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*, (2002) pp. 175–189.

[10] Harel, A.; Shabtai, A.; Rokach, L.; Elovici, Y., "M-Score: A Misuseability Weight Measure," Dependable and Secure Computing, *IEEE Transactions*, vol.9, no.3, pp.414, 428, May-June, DOI: 10.1109/TDSC.2012.17, (2012)

[11] Yaseen, Q., Panda, B., Insider threat mitigation: preventing unauthorized knowledge acquisition. *International Journal of Information Security,Springer-Verlag*. vol. 11, P 269-280. DOI: 10.1007/s10207-012-0165-6, (2012)

[12] Zhong, Y., Qin, X., Database Intrusion Detection Based on User Query Frequent Itemsets Mining with Item Constraints. *In Proceeding of the 3rd international conference on information security*, (2004) pp. 224–225.

[13] Bertino, E., Terzi, E., Kamra, A., Vakali, A., Intrusion Detection in RBAC-Administered Databases. *In Proceedings of the 21st annual computer security applications conference (ACSAC)*, (2005) pp. 170–182.

[14] Hu, Y., Panda, B., A Data Mining Approach for Database Intrusion Detection. *In Proceedings of the ACM symposium on applied computing*, (2004) pp. 711–716.

[15] Srivastava, A., Sural, S., Majumdar, A.K., Weighted Intratransactional Rule Mining for Database Intrusion Detection. *In Proceedings of the Pacific-Asia knowledge discovery and data mining (PAKDD)*, lecture notes in artificial intelligence, Springer, (2006) pp. 611–620.

[16] Lin, X., Mukherji, A., Rundensteiner, E. A., Ruiz, C., Matthew O. Ward, M. O., PARAS: a parameter space framework for online association mining. *Proc. VLDB Endow. 6, 3 (January 2013),* 193-204. DOI=10.14778/2535569.2448953, (2013)

[17] Adinehnia R., Udzir N. I., Suriani L., Affendey I. I., Zurina M. H., Effective Mining on Large Databases for Intrusion Detection. *International Symposium on Biometrics and Security Technologies (ISBAST14),* Kuala Lumpur, Malaysia, (2014)

[18] Agrawal, R., Imielinski, T, & Swami, A., Mining Association Rules Between Sets of Items in Large Databases. *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, Washington DC, USA, CM Press 1993: 207 - 216. (1993)

[19] Hipp, J., Guntzer, U., & Nakhaeizadeh, G., Algorithms for association rule mining — a general survey and comparison. *ACM SIGKDD Explorations Newsletter* 2(1): 58-64. (2000)

[20] Zachary K. Baker , Viktor K. Prasanna. Efficient Parallel Data Mining with the Apriori Algorithm on FPGAs. *In Submitted to the IEEE International Parallel and Distributed Processing Symposium (IPDPS '05),* (2005)

[21] Agrawal, R. & Srikant, R..Fast Algorithms for Mining Association Rules in Large Databases. *In Proceedings of the 20th international Conference on Very Large Databases (September 12 - 15, 1994),* (1994)

[22] Domingos, P., Pazzani, M.J., On the optimality of the simple bayesian classifier under zero-one loss. Mach. Learn. 29(2–3), 103–130. (1997)

[23] Mitchell TM., Machine Learning. McGraw-Hill, New york. (1997)