Intelligent Multi-Agent Based Multivariate Statistical Framework for Database Intrusion Prevention System

P. Ramasubramanian and A. Kannan
School of Computer Science and Engineering, Anna University, India

Abstract: *This paper describes a framework for highly distributed real-time monitoring approach to database security using intelligent multi-agents. The intrusion prevention system described in this paper uses a combination of both statistical anomaly prevention and rule based misuse prevention in order to detect a misuser. This paper describes a framework for a statistical anomaly prediction system using a multivariate statistical forecasting model, which predicts unauthorized invasions of user based on previous observations and takes further action before intrusion occurs. This paper focuses on detecting significant changes of transaction intensity for intrusion prevention. The experimental study is performed using real data provided by a major Corporate Bank. Furthermore, a comparative evaluation of the proposed model over the traditional statistical forecasting models was carried out using mean absolute percentage error on a prediction data set and a better prediction accuracy has been observed. The misuse prevention system uses a set of rules that define typical illegal user behavior. A separate rule subsystem is designed for this misuse detection system and it is known as Temporal Authorization Rule Markup Language (TARML). In order to reduce single point of failures in centralized security system, a dynamic distributed system has been designed in which the security management task is distributed across the network using intelligent multi-agents.*

Keywords: *Multi-agents, database security, statistical database anomaly prediction, database misuse detection.*

Received April 15, 2004 ; accepted August 16, 2004

1. Introduction

In today's business world, information is the most valuable asset of organizations and thus requires appropriate management and protection. As organizations are increasing their reliance on the distributed computing environment are becoming more vulnerable to security breaches. Any breach of security to these databases can result in tarnished reputation for the organization, loss of customer's confidence and might even result in lawsuits [6]. Many other mechanisms and technologies like firewalls, encryption, authentication, vulnerability checking, access control policies can offer security but it is still susceptible for attacks from hackers who takes advantage of system flaws and social engineering tricks. In addition, computer systems with no connection to public networks remain vulnerable to disgruntled employees or other insiders who misuse their privileges [14]. This observation results in the fact that much more emphasis has to be placed on internal control mechanisms of systems like audit log analysis.

In information systems, the primary security threat comes from insider abuse and from intrusion. Security policies do not sufficiently guard data stored in a database system against "privileged users". Many intrusions into information systems manifest through the significantly increased or decreased intensity of transactions occurring in information systems [17]. For example, intruders who have gained super-user privileges can perform malicious transactions and disable many resources in the information system, resulting in the abruptly decreased intensity of transactions. This reinforces the point that intrusion detection systems should not only be employed at the network and hosts, but also at the database systems where the critical information assets lie [6]. Therefore, the early detection of significant changes in the transaction intensity can help stop many intrusions early to protect information systems and assure reliability of information systems.

The remainder of this paper is organized as follows. Section 2 provides an overview of related works on intrusion detection and makes a comparison with our model. Section 3 presents a short description of audit metrics and user profile that are used to perform database anomaly prediction. In section 4, we deal with the design of the BayTLsq prediction algorithm that is used for performing the statistical anomaly method as a prediction task. In section 5, the design of a distributed architecture to support database security system has been described. Section 6 explains the different phases of experiments that were taken in our BayTLsq forecasting model and also presents the experimental results obtained using real data provided by a major

Corporate Bank, Chennai. It also focuses on the performance analysis of BayTLsq model with the traditional statistical prediction models. Finally, section 8 provides a concluding remark and suggests some further enhancements.

2. Related Works

The existing intrusion detection systems operate in real time, capturing the intruder when or after intrusion occurs. From the existing methods of detecting the intrusion [7, 8, 9], we observed that all intrusion detection systems were lacking a vital component: that they take action, after the intrusion has been detected [16]. This serious weakness has led to the research on forecasting models. However, though the Intrusion Detection system is real-time, it can detect the intrusion after the action, but never before [10]. To address the problem of detecting intrusions after they take place, we utilize a multivariate statistical prediction algorithm, which takes into account user behaviour and generates a predicted profile to foresee the future user actions.

Intrusion detection research is not new and has been on going for many years. However, previous works were focused largely on network-based intrusion detection [9, 10, 17] and host-based intrusion detection [18]. These intrusion detection systems do not work at the application layer, which can potentially offer more accurate and precise detection for the targeted application. The distinctive characteristics of database management systems, together with their widespread use and the invaluable data they hold make it vital to detect any intrusion attempts made at the databases. Therefore, intrusion detection models and techniques specially designed for databases are becoming imperative needs [6]. Most of the research on database security revolves around access policies, roles, administration procedures, physical security, security models and data inference. Little amount of work is done on database IDSs even though most emphasis in literature has been found for Network IDSs [10]. DIDAFIT [8] is a database misuse detection system that identifies anomalous database accesses by matching SQL statements with a known set of legitimate database transaction fingerprints. But the key drawback of this misuse detection approach is that they cannot detect novel attacks against systems that leave different signatures. So, the rate of missed attacks (false negatives) can be extremely high depending on the ingenuity of the attackers. In Chung *et. al* [3] work, a method was devised which generates profiles of the users and their roles in a relational database system. This method assumes that the legitimate users show some level of consistency in using the database system. If this assumption does not hold, or if the threshold for inconsistency is not set properly, the result will be a high level of false positives. It also faces the attribute selection problem like choosing a feature in building a work scope [6]. To the best of the author's knowledge, there is no report on an intrusion prevention system for databases. As far as the authors know, this is the only work using statistical forecasting model to predict database intrusions.

The process of monitoring user behaviour and making predictions on these data is a linear problem and thus uses statistical models rather than neural networks version [16]. Artificial neural networks (ANNs) have been widely applied to short term forecast problems [14]. These models, however, have their limitations owing to the tremendous noise and complex dimensionality of data besides, the quantity of data itself may also interfere with the learning of patterns. So, ANNs are rarely used to do real-time predictions. The main disadvantage of using neural networks is that they pose a stability-plasticity dilemma, i. e., once if they are trained and later if they learn new patterns (plasticity) they fail to remember previous knowledge (stability). Thus, they are not adequate to offer an adaptive solution [13]. Since they are very flexible, they find many false patterns in a low signal to- noise ratio situation. In addition, the knowledge contained in their weights cannot be expressed in human understandable terms [10]. Moreover, neural networks require an enormous amount of observed data to acquire better approach, so that these models probably are not suitable for short-term forecasting because in short-term forecasting only a small number of data's are considered. Thus, this paper uses a short-term linear statistical method to solve the prediction problem. Typically, statistical models, like Simple Exponential, Holt-Winters smoothing, Regression, Grey, Bayesian and Box-Jenkins are widely used in the literature [16]. However, Grey, Holt-Winters Smoothing, Regression method and Box-Jenkins, usually require many observed data to obtain better approach, so these models are also not suitable for short-term forecasting. Secondly, Holt-Winters Smoothing and Regression methods are applicable only for seasonal or cyclical data series [13]. Contrarily, the Bayesian model requires a few sampled points for prediction would achieve the better prediction accuracy.

Pikoulas *et al.* [10] work uses Bayesian forecasting model to detect future intrusions against systems. Their technique was based on statistical method and there is still a need for improvement of the accuracy of the prediction method. Although the Bayesian model has the advantage of simple and fast to predict the future output, the precision is also still arguable since it encounters the problem in which the predicted results cannot reach a satisfactory need because the overshooting predicted value from the Bayesian prediction model will turn out to be an overestimated (or underestimated) result at the position of turning

points and causes big residual errors at the turning points where the peak or valley observed values occurred [16]. The cumulative temporal least squared linear model have the problem about the damped around turning points, that is conversely to the situation happened to Bayesian model [2]. Therefore, we can apply this characteristic to offset the magnitude overshooting such that alleviating the effect of the magnitude of the original given data for Bayesian model can be achieved.

3. User Audit Profile

Our intrusion prevention system uses hybrid technique. Thus, the user profile is a collection of real-time negative authorization rules stated by database administrators and audit record. The rules include the access of database objects of the network computer system for which permission is not granted, data objects that users cannot use on their hosts, and even includes privileges that the database administrators feel that the users should not use.

3.1. Temporal Authorization Rule Markup Language

In our model, a negative authorization is specified as (time, auth), where time is a temporal attribute, and auth (s, o, p) is an authorization. Here, temporal represents either valid time or transaction time, during which auth is invalid. s represents the subject, o represents the database object and p, the privilege. These rules are represented by means of Event-Time-Condition-Action (ETCA) [11] rules. Stating the purpose of ETCA rules briefly, whenever the event takes place the negative authorization condition corresponding to it's checked and if the condition is satisfied then the defence action to be performed on a user when an attack signature is detected.

3.2. Audit Record

Auditing is used to investigate suspicious activity. There are three standard types of auditing to monitor the user behaviour namely SQL statement-level, privilege-level and object-level auditing. Statement and privilege audit options are in effect at the time a database user connects to the database and remain in effect for the duration of the session. In contrast, changes to object audit options become effective for current sessions immediately [14]. So in this work, we have chosen object level auditing to build profiles. The object-level auditing can be done by user on successful or non-successful attempts for session intervals. A session is the time between when a user connects to and disconnects from a database object. We need a utility to capture the submitted database transactions in order to compare them with those in the legitimate user profile. Oracle provides the sql trace [8] utility that can be used to trace all database operations in a database session of a user. We make use of its capability to log SQL transactions executed by the database engine.

4. Prediction Algorithm

BayTLsq forecasting model makes periodic short-term forecasts, since long-term forecasts cannot accurately predict an intrusion [16]. In this we use a multivariate time series technique to forecast the hacker's behaviour effectively.

4.1. Bayesian Forecasting Model

Consider the two basic equations

- The observation equation

$$Y_t = F_t^T \theta_t + v_t,\ v_t\ N\ [0, \sigma]$$

- The system or transition equation

$$\theta_t = \theta_{t-1} + \omega_t,\ \omega_t\ N\ [0, \sigma W_t]$$

Where:
Y_t: is a variable that we observe at time t.
F_t: is an n-dimensional vector.
θ_t: is an n-dimensional parameter vector.
v_t: is a random variable that has a normal distribution with zero mean and variance σ.
ω_t: is an n-dimensional random vector with a multivariate normal distribution with zero mean and variance matrix σW_t.

To calculate the predicted value for time t we need to calculate f_t first, which is $F_t m_{t-1}$, and then we have to calculate Y_t. The Mean $m_{t-1} = E(\theta_{t-1} | D_{t-1})$, where E is the Expectation of the θ_{t-1} given D_{t-1}. We also need to calculate the variance C_{t-1} which is $C_{t-1} = Var(\theta_{t-1} | D_{t-1})$, where Var is variance.

4.2. Temporal Least Square Prediction Model

Let r (a_1, a_2, ..., a_n, t_1, t_2, t_3) be a relation in intelligent temporal database where r is the relation name, a_1, a_2, ...,a_n are attributes of r and each a_i has attribute values x_1, x_2, ..., x_m where m N and t_1, t_2 and t_3 are the temporal attributes. Each t_i has values of the form y_1, y_2,....y_m such that for each y_i called the start time, there exists an y_j called end time representing the interval [y_i, y_j] during which the attribute value is valid.
The set of points (x_i, $_{yi}$) can form a straight line, parabola or a curve of a polynomial of degree k. Using the method of Least Squares; a suitable curve is fitted for these points. If $y = a_0 + a_1x + a_2x^2 + ... + a_kx^k$ is the k^{th} degree polynomial of best fit to the set of points (x_i, y_i), i = 1, 2, ..., n, the constants a_0, a_1, ..., a_n can be obtained by solving the equations:

$$\Sigma y_i = na_0 + a_1\Sigma x_i + a_2\Sigma x_i^2 + ... + a_k\Sigma x_i^k$$
$$\Sigma x_i y_i = a_0 x_i + a_1\Sigma x_i^2 + a_2\Sigma x_i^3 + ... + a_k\Sigma x_i^{k+1}$$
$$\Sigma x_i k y_i = a_0\Sigma x_i^k + a_1\Sigma x_i^{k+1} + a_2\Sigma x_i^{k+2} + ... + a_k\Sigma x_i^{2k}$$

Where each summation extends over i from 1 to n. For each pair of temporal and non-temporal attributes specified, there exists a mapping from the temporal attribute to the non-temporal attribute specifying the period of validity of the non-temporal attribute values.

4.3. BayTLsq Prediction Algorithm

All simulations were run under R [4] for Windows on a 2GHz computer with 512 MB of RAM. Bayesian and Temporal Least Squared Linear Model functions were programmed in C by the authors.

A temporal least squared linear model combining with Bayesian model thus is exploited for the prediction as follows.

$$bpt = \frac{w_1 * Y_t + w_2 * z_t}{2} \quad (2)$$

Where Y_t and z_t stand for the predicted value of a Bayesian model and the predicted value of a temporal least squared linear model, respectively; moreover, the w_1 and w_2 represent the weight of Y_t and z_t, respectively.

The value of w_1 and w_2 can be evaluated by a weighting algorithm [2] where r (t) represents the predicted output from a temporal least squared linear model.

5. Architecture

The general architectural framework for a Multi-Agent based database intrusion prevention system is illustrated in Figure 1. It has been implemented by using Aglets Software Development Kit (ASDK) [5], and API Java Aglet (J-AAPI) developed by IBM Tokyo Research Laboratory. In this architecture, two kinds of agents are considered:

1. Information Agent
2. Host Agent.

5.1. Information Agent

Information Agent (Static Agent) acts as a data processing unit, and as a data repository for the Host Agents. It is responsible for collecting and storing user profiles for all users from various agents in a timely fashion that has access to the data in the protected network. Also it provides the user profile to the Host Agent whenever it is requested. The Information Agent comprises of three main components namely 1. Host Monitor 2. XML Audit Profile Server, and 3. Admin Interface:

1. *Host Monitor (Mobile Agent):* In distributed environment, the performance of each host has to be monitored constantly so that performance drop or failure of any node can be detected. Based on that corrective measures can be taken to maintain the overall performance level of the network. When an Information Agent is created, it sends a monitor agent to every host in the network. The monitor agent then starts monitoring the performance as soon as it reaches the host at regular intervals and this interval can be programmed.
2. *XML Audit Profile Server*: Audit records are written into the XML format and they are stored in XML Audit Profile Server. The generation and insertion of an audit trail record is independent of a user's transaction. Therefore, even if a user's transaction is rolled back, the audit trail record remains committed. The profile server must be able to provide the user behaviour information about past, present and future and it must allow forecasting based on temporal logic [12].

   So, in this work we have chosen an audit record database, which maintains past, present and future data about users and is termed as a temporal database. Authorization rules are installed in the XML repository and these rules monitor the XML databases for the occurrence of events by the construction of event listeners on each node of the XML document.
3. *Admin Interface:* Interface agent (Static Agent) provides friendly human-computer interface for system administrator and it can provide information for administrator in the form of GUI and receive control commands from the GUI.

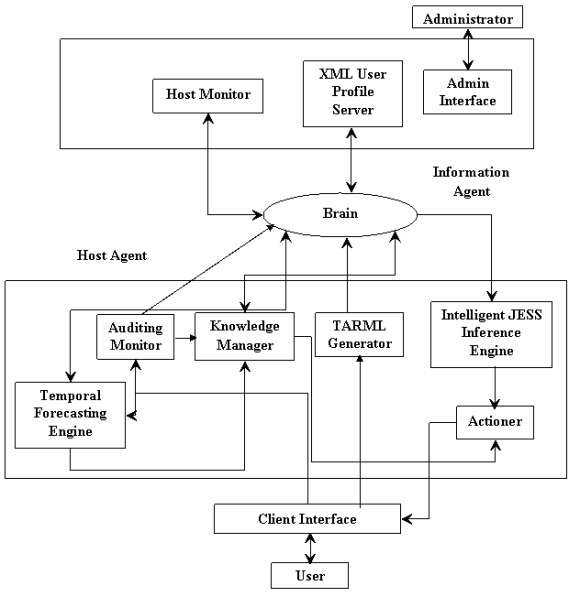


Figure 1. Intelligent multi-agent based database statistical anomaly prediction system.

5.2. Brain (Static Agent)

Brain (Static Agent) is the central component of the agent and it initiates, controls, coordinates and integrates the activities of all the components of both Information Agent and Host Agent.

5.3. Host Agent

A Host Agent resides on every host on the protected distributed database environment. It can be split into three basic intelligent agents such as auditing monitor, knowledge manager, and actioner:

- *Auditing Monitor*: This static agent monitors every user who logs into the system. The database objects, privileges of the current users on the host machine are logged and send to the information agent.
- *Temporal Forecasting Engine*: It is responsible for processing the monitored data from the Host Agent and generates forecasting data for the next session of the specific user. We move the forecasting module from the Information Agent to the Host Agent and repeat the experiments to discover if there is any difference in the results, and to maximize the intrusion forbidden system performance, so the agents will be distributing not only the security, but also the workload of the processing requirements of the Information Agent.
- *Profile Reader*: This mobile agent is responsible for fetching the on-line data from the auditing monitor and the predicted values from the temporal forecasting engine. Then it sends this information to the Information agent.
- *Rule Generator*: This static agent is assigned the task of rule creation based on the request from the client and it is responsible for the submission of the rules to the XML Server.
- *Intelligent JESS Inference Engine*: When a transaction like insertion of element, deletion of element or updating of element happens at the XML file, the ETCA rule given in XML format is mapped to JESS authorization rule, which is predefined by database administrator. The events that occur are also converted to JESS facts. The JESS Inference engine constantly monitors the JESS rule and on the occurrence of a new fact, executes the JESS rule producing new JESS facts as the result. These JESS facts must later be converted into suitable action and it's transmitted to the client. It checks predefined rules in order to detect database anomalies caused by successful attacks. Users are then granted access privileges to the system containing data only, which they have been authorized via a JESS Rule Execution Engine. If the JESS Inference engine fires the rule that gets reflected in the database dynamically. This is happened dynamically when the client is still in the transaction.
- *Knowledge Manager*: This mobile agent gets the appropriate profile for the specific user, which is stored locally on the XML Audit Profile Server, and then it compares the user's historical profile with the information sent by the Host Monitor. The knowledge manager makes comparison constantly. If the current behaviour profile does not match with the normal behaviour pattern defined by the user historical profile, then the Knowledge Manager provides the following information to the Actioner: user identifier, session identifier, host identifier & IP address and the invalid privilege with the corresponding unauthorized object attempting to be accessed.
- *Actioner*: Actioner's (Static Agent) role is to take necessary actions when an intrusion is detected. It also uses the prediction data for a user, to take preemptive actions on the user behaviour. When an attack is detected exactly, the Actioner does one of the following operations to terminate the attack: 1. Reject the user's attempt with the warning message 2. Terminate the specific operation on the particular database object 3. Lock the user's keyboard and prevent the user from consuming any further data resources 4. Reports an intrusion detected on a host to the system administrator via the Information Agent. In Actioner, the action element is put in the SOAP [1] component. Then the SOAP header and SOAP envelope are constructed over it by putting the endpoint of where the data has to be sent. The output of this component is SOAP message, which is then appended to the action part of the rule. Java API for XML Messaging (AXM) is a package that is used to send the SOAP message across different clients. The result should be presented in the user readable/understandable form. For example, XML documents can be presented as HTML pages with XSLT style sheets.
- *Client Interface*: It is just an application dependent program that communicates with the client to get the client's request and also provides the response to the client. In case of Stationary systems, the system needs to communicate via a network using standard HTTP format. In order to support mobile users, this component converts the XML data into Wireless Markup Language (WML) [15] data and a Wireless Application Protocol (WAP) [15] is used to transfer this WML data to the mobile devices.

6. Experimental Results

6.1. Model Development and Analysis of Data

This study obtains a collection of audit data for normal transactions from a major corporate bank, Chennai, India. The database objects considered are:

1. Customer Deposit Accounts (CDAcc).
2. Customer Loan Accounts (CLAcc)
3. Ledger Reports related to each transactions on the Customer Accounts (LRep).

The database objects are used by Tellers (Tlr), Customer Service Reps (CSR) and Loan Officers (LO) to perform various transactions. It is also used by Accountants (Acc), Accounting Managers (AccMr) and Internal Auditors (IntAud) to post, generate and

verify accounting data. Branch Manager (BrM) has the ability to perform any of the functions of other roles in times of emergency and to view all transactions, account statuses and validation flags. Normal transactions are generated by simulating activities observed in a corporate bank information system in an usual operation condition. A number of intrusions are also simulated in our laboratory, including password guessing, to gain the root privilege attempts to gain an unauthorized remote access, an overwhelming number of service requests can be sent to an information system over a short period of time to deplete the computational resource in the server and thus deny the server's ability to respond to user's service requests, etc., to create the audit data of intrusive activities.

6.2. Selection of Model Inputs and Architecture

We designed BayTLsq model to predict transaction intensity rate for the next minute for the current user based upon the following input variables: 1. User ID 2. Object ID 3. Privilege ID 4. Session ID 5. User Authorization level (High, Medium, Low) 6. Hours off (% of total) 7. Days off (% of total) like holidays and weekend 8. Hour-of-day 9. Day-of-week 10. Historical transaction rate: command rate/minute of the previous four weeks were collected and used as historical data inputs; these data reflect the user habit of consuming resource.

6.3. Training and Testing Data

We obtain 8 weeks of the December 2003 & January 2004 audit dataset from the corporate bank in our study. We use the first part of the audit data for normal activities as our training dataset, and use the remaining audit data for normal activities and attack activities as our testing dataset. The first half of the audit data, consisting of 16,413 audit transactions lasting four weeks, is used as the training data. In the testing dataset, the average session length is comparatively smaller in week-2 and week-3 than that in week-1 and week-4. In terms of sessions, almost one-fifth of the sessions in week-2 and one-fifteenth of the sessions in week-3 are intrusion sessions. Week-1 contains mostly normal sessions, week-4 also does not have too many intrusion sessions. Week-1 and week-2 contain 12 and 16 normal sessions, week-2 and week-3 contain 6 and 7 intrusion sessions. Hence, the testing data contains a total of 28,574 audit transactions with 3 segments of data in the sequence: 1. 7,320 normal events (the first half of the 14,640 normal events) 2. 13,934 intrusive events 3. 7,320 normal events (the second half of the 14,640 normal events).

6.4. Operation Stages

Our model has three phases of operations:

- *Training phase*: For each audit event in the training dataset, first obtain the average transaction-intensity of a user for a session by computing the number of transactions for each second. Interference mechanism is used, if new users log-on to the system or when we don't have sufficient number of observations for users. It provides enough information about users transactions for prediction, which we can then add them to the forecasting model prior the prediction.
- *Forecasting phase*: During the forecasting phase, the system computes a single-step ahead of the transaction-intensity prediction profile.
- *Testing Phase*: For each audit event in the testing data set, first compute the mean and standard deviation of predicted values. The prediction signals thresholds to detect anomalies are usually set to = [Mean + 3 x Standard Deviation, Mean - 3 x Standard Deviation] [16]. If the predicted value for an observation falls outside the lower and upper threshold, an anomaly is detected and an alarm signal is generated.

6.5. Slope Results

One comparison of the two values that we are getting every time t is to compare the slope of the two graphs. In this way, even if the values do not match, if their slopes are close together, we have a very strong indication as to what behaviour the user will have. The slope is a very useful tool to compare our results with the actual values [16]. The meaning behind the slope is that if we observe the slope of each prediction, we can predict if the user is likely to use the data resource that we are monitoring, or not. The prediction does not have to match exactly the value of the real observation, but to have the same slope as the real observation. Figure 2 outlines the slope results of these tests.

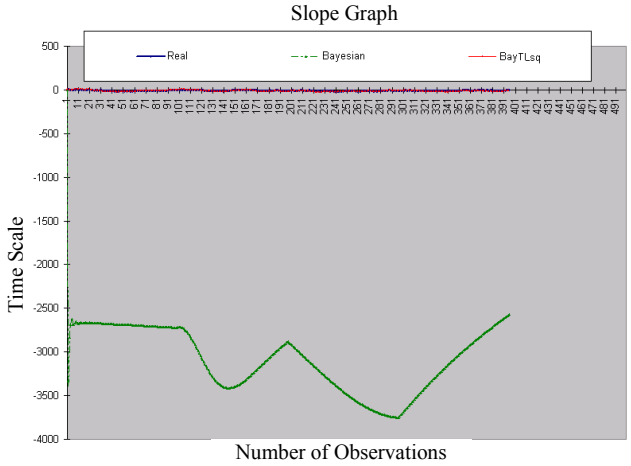


Figure 2. Slope graph comparison between the two results.

From Figure 2, we can observe that the two lines are following each other and BayCLsq model is able to predict the deviations of the user behaviour by keeping

the changes of the user. By comparing the slope, for the graph for the same period, from the real values and the prediction values, we can see that both slopes are the same, or similar, which is enough to make a prediction. As we can see in Figure 2, that the two lines that represent the real observations and the predicted ones, are not exactly matching. However, we can observe that if we get the any two pair of points from both graphs and calculate their slope, we can see that it is very similar (Figure 2). This shows that even if our BayTLsq prediction model can not some times predict how long the user will use a specific data resource in the future, it can predict if the user is going to use this specific data resource or not.

6.6. Quantile-Quantile Plot

Another analysis tool that was used was the Quantile-Quantile (QQ) plot. It is a graphical technique for determining if two data sets come from populations with a common distribution. A 45-degree reference line is also plotted. If the two datasets come from a population with the same distribution, the points should fall approximately along this reference line.

The greater the departure from this reference line, the greater the evidence for the conclusion that the two data sets have come from populations with different distributions.

It can be seen that the BayTLsq model could represent most of the variability within the dataset (Figure 3).

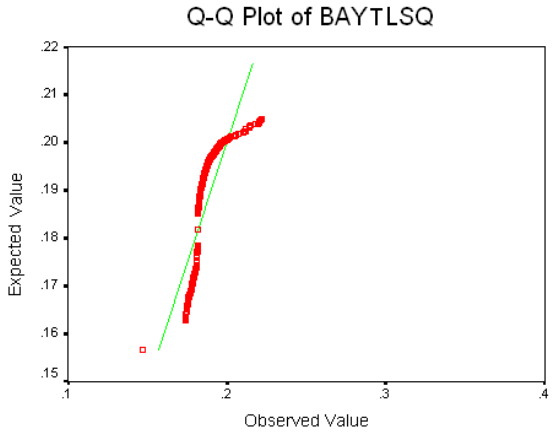


Figure 3. QQ plot for the BayTLsq model.

7. Performance Analysis Results

In order to evaluate the accuracy of our forecasting model, a widely accepted quantitative measure, such as Mean Absolute Percentage Error (MAPE) has been used. The performance analysis of the BayTLsq forecaster was measured in terms of Mean Absolute Percentage Error (MAPE) =

$$\frac{1}{N}\sum_{i=1}^{N}\left|\frac{Forecast - Target_i}{Target_i}\right|$$

Where N = Number of observations. MAPE will be employed in this paper as the performance criterion, for its easy understanding and simple mathematical computation. The mean absolute percentage error determines the mean percentage deviation of the predicted outputs from the target outputs. This absolute deviation places greater emphasis on errors occurring with small target values as opposed to those of larger target values [13]. Error statistics are shown for the normal (training) set in Table 1.

Table 1. Error measurement.

Models	Tlr	CSR	LO	Acc	AccMr	IntAud
BayTLsq	0.2792	0.6230	0.0713	1.6198	2.6445	2.8624
TLSq	2.0667	2.9716	1.8073	4.2297	4.5408	7.3359
Bayesian	2.4115	3.2425	1.9102	4.2444	5.8624	8.2338
Grey	3.2786	5.7335	2.1208	4.4043	5.9016	8.7680
Holt-Winters	5.1448	7.6308	4.1392	7.8316	18.8750	48.4673
Box-Jenkins	16.5670	18.003	8.2587	10.407	39.695	70.72

While the most accurate predictions are achieved similarly with Bayesian and Grey, the former requires small amount of observations for making predictions and on-line learning capabilities that make Bayesian a better choice. The only difficulty with the Bayesian model is the specification of the initial values, such that the algorithm may be put into practice [16]. Although speed and simplicity were not considered important, the temporal least squared linear regression model was the simplest (no parameters to adjust) and the fastest. To build the Box-Jenkins model we did not need much computation, but rather human effort and heuristic decisions to determine the model parameters [13]. In addition, its best performance is worse than that achieved by any of the statistical models. This can be due to the importance of days off, that introduces irregularities in the time series.

8. Conclusions and Future Work

In this paper, an intelligent multi-agent based distributed database intrusion prevention system has been presented to learn previously observed user behaviour in order to prevent future intrusions in database systems. In this paper, a statistical database anomaly intrusion prediction system has been presented to learn previously observed user behaviour in order to prevent future intrusions in database systems and a new way to improve false-alarm detection using a BayTLsq forecasting model. The results from our technique show the viability of our approach for detecting intrusions. The comparison results show that our model works very well and outperforms the traditional statistical forecasting models, especially in the short-term period. The TARML has been designed in misuse prevention system and it can suit for any kind of real domain and any platform. For future expansion fuzzy rules can be

extended with JESS, so that the intrusion prevention system can be made much more effective. Thus, the system is developed to demonstrate the use of intelligent agents for auditing the transactions within the organization, detecting potential risks, and avoiding uncontrollable transactions. In fact, more advanced intervention analysis can be incorporated which would, most likely, produce even better results. Hence in the future works, we'll research on the intelligent method of intrusion detection further to improve the predictability of database intrusion and reduce the rate of false negative alarm and false positive alarm.

Acknowledgments

The authors gratefully acknowledge the anonymous reviewers for their useful comments.

References

[1] Box D., Ehnebuske D., Kakivaya G., Layman A., Mendelsohn N., Nielsen N. F., Thatte S., and Winer D., Simple Object Access Protocol (SOAP) 1.1., http://www.w3.org/TR/2000/NOTE SOAP-20000508, http://www.w3.org/TR/2000/S OAP, 2004.

[2] Chang B. R. and Tsai S. F., "An Intelligent Prediction Method for Short-Term Time Series Forecast on Engineering Education," *in Proceedings of the International Conference on Engineering Education*, Manchester, UK, 2002.

[3] Chung C. Y., Gertz M., and Levitt K., "Misuse Detection in Database Systems Through User Profling ," *in Proceedings of the 2nd International Workshop on the Recent Advances in Intrusion Detection (RAID)*, pp. 278, West Lafayette, September 1999.

[4] Imai K., King G., and Lau O., Statistical R for Windows (1.9.0), URL http://cran.r-project.org, 2004.

[5] Java Aglet, IBM Tokyo Research Laboratory, http://www.trl.ibm.co.jp/aglets, 2004.

[6] Lee S. Y., Low W. L., and Wong P. Y., "Learning Fingerprints for a Database Intrusion Detection System," *in Proceedings of the 7th European Symposium on Research in Computer Security*, Zurich, Switzerland, pp. 264-280, 2002.

[7] Lee W. and Stolfo S. J., "A Framework for Constructing Features and Models for Intrusion Detection Systems," *ACM Transactions on Information and System Security*, vol. 3, no. 4, pp. 227-261, November 2000.

[8] Low W. L., Lee J., and Teoh P., "Didafit: Detecting Intrusions in Databases through Fingerprinting Transactions," *in Proceedings of the 4th International Conference on Enterprise Information Systems*, Ciudad Real, Spain, pp. 121-128, April 2002.

[9] Michael C. C. and Ghosh A., "Simple State-Based Approaches to Program-Based Anomaly Detection," *ACM Transactions on Information and System Security*, vol. 5, no. 3, pp. 203-237, 2002.

[10] Pikoulas J., Buchanan W. J., Manion M., and Triantafyllopoulos K., "An Intelligent Agent Intrusion System," *in Proceedings of the 9th IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS)*, IEEE Computer Society., Luden, Sweden, pp. 94-102, 2002.

[11] Ramasubramanian P. and Kannan A., "An Active Rule Based Approach to Database Security in Ecommerce Systems Using Temporal Constraints," *IEEE Tencon'2003*, Bangalore, India, pp. 1047-1053, October 2003.

[12] Ramasubramanian P. and Kannan A., "Interfacing an Efficient and Portable Natural Language Query Interface to Temporal Databases," *International Journal of Information Technology*, vol. 10, no. 1, pp. 88-100.

[13] Ramasubramanian P. and Kannan A., "Multivariate Statistical Short-Term Hybrid Prediction Modeling for Database Anomaly Intrusion Prediction System," *in Proceedings of the Second International Conference on Applied Cryptography and Network Security (ACNS'2004)*, Yellow Mountain, China, June 2004.

[14] Ramasubramanian P. and Kannan A., "Quickprop Neural Network Short-Term Forecasting Framework for a Database Intrusion Prediction System," *in Proceedings of the 7th International Conference on Artificial Intelligence and Soft Computing (ICAISC'2004)*, Zakopane, Poland, Springer-Verlag, vol. 3070, pp. 847-852, June 2004.

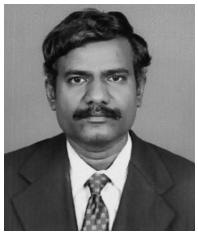
[15] Syvnen A., WAP Specifications and WAP Gateway, http://www.wapforum.org, http://www. wapgateway.org, 2004.

[16] Triantafyllopoulos K. and Pikoulas J., "Multivariate Bayesian Regression Applied to the Problem of Network Security," *Journal of Forecasting*, vol. 21, pp. 579-594, 2002.

[17] Ye N. and Chen Q., "An Anomaly Detection Technique Based on A Chi-Square Statistic for Detecting Intrusions into Information Systems," *Quality and Reliability Engineering International*, vol. 17, no. 2, pp. 105-112, March/April 2001.

[18] Ye N., Vilbert S., and Chen Q., "Computer Intrusion Detection through EWMA for Auto correlated and Uncorrelated Data," *IEEE Transactions on Reliability*, vol. 52, no. 1, pp. 75-82, 2003.

P. Ramasubramanian is currently pursuing his PhD degree in computer science and engineering at Anna University, Chennai, India. He received the Bachelor of engineering and Master of engineering in computer science and engineering from Madurai Kamaraj University, India in 1999 and 2001, respectively. His research interests include network, database security, soft computing, and machine learning.



A. Kannan received his Master of engineering and PhD degrees in computer science and engineering from Anna University, India, in 1991 and 2000, respectively. Currently, he is an assistant professor in the Department of Computer Science and Engineering at Anna University, India. His research interests include databases systems, software engineering, and soft computing.