Behavior Reliance Anomaly Detection with Customized Compact Prediction Trees

K. Venkateswara Rao, T. Uma Devi

Abstract: ***Since beginning of the cloud computing, service providers are always concentrated to protect their cloud technical assets from external adversaries and neglected the insider threat mitigation activities, caused to remain the today clouds as vulnerable to malicious insider threats. Malicious insiders are the integral part of our cloud environment with authentic access to critical app modules, data, assets and other objects, so they are insidious. Several threat detection models like behavior based, rule-based, activity based and impersonation based were introduced by former research scholars to find the malicious insiders from cloud environment. The major and common limitations identified from the former malicious insider threat detection models are 1) uncertain prevention from insider threats2) over load on cloud servers due to threat detection process3) suffering from false negatives in results and 4) none of the threat detection models were comprehensive. This paper proposed the Behavior Reliance Anomaly Detection (BRAD) approach, to find the malicious insiders precisely and to address the aforementioned insider threat detection limitations with cutting-edge technologies. Experimental results with BRAD prototype specified that, the proposed BRAD is proved as reliable, scalable and comprehensive than the former behavior based anomaly detection.***

Index Terms: Cloud Security, Malicious Insider Threats, Behavioral Analysis, Compact Prediction Tree (CPT), Anomaly Detection Algorithm.

I. INTRODUCTION

Today Malicious Insider threat became a burning security issue in the area of cloud computing notified by Cloud Security Alliance [1]. Ponemon Institute conducted survey[2] revealed that, top most US based Cloud or enterprise organizations are spending nearly $9 million per year, to identify and mitigate the malicious insider threats, to protect their clouds from insider threats. The total 159 organizations which are participated in their survey are the victims of malicious insider threat in some way. The results of Ponemon survey revealed that, only the intended malicious insiders and credential based attackers bagged 36% of total insider attacks, which are happening due to the lack of monitoring and surveillance on admin activities. Finally the survey concluded that most of their participant organizations recognized the insider attacks after the long time since they happen. In some cases these insider attacks were not even recognized by the organizations till today, due to lack of the research, awareness and negligence.

Since a decade, malicious insiders are updated and explored their activities to perform different types of attacks on cloud environment are: spoofing of identity, data tampering, information disclosure, espionage and sabotage. Rule based detection, activity based detection, behavioral detection, Technical detection and impersonation based detection are the popular insider thereat detection mechanisms [4 and 5]. Cloud Admins, System Admins, Data Admins, Network Admins, Process flow admins, Business Partners, Former employees, Testers and Application Managers are some example job roles (as shown in figure-1.) [3], which are frequently appearing in identified malicious attackers list. These job roles are the authorized administrators of cloud environment and each job role is assigned with their job relevant activities, boundaries and responsibilities. In general these admins are issued with the high level privileges to perform the custom activities (i.e. CRUD operations) on system data as they needed, hence there may be a chance to misuse these privileges by admins for their personal benefits. Ponemen survey reports [2] and Kandias et al [3] conducted analysis revealed that, the unsatisfied employees, frustrated employees, business rival agents, crime background employees and some others are becoming as malicious insiders. In order to prevent the cloud environment from malicious insider activities, several insider activity identification mechanisms (behavior-based, rule-based, activity-based and impersonation-based etc.) have been proposed by the earlier research scholars. Rule based detection model defines a set of rules (as specification) for each job role activity and these rules are compared against all activities of that job role. This model believes that, any violation of rules by the job role activities indicate that some suspicious behavior, which may cause to an insider threat. This model recorded high accuracy in detecting the malicious insiders while doing data exfiltration [3].In this case, thousands of rules have to be selected and verified against each request (activity) made by the job role. This is a tedious time complex process and also consumes the huge amount of storage resources along with processor time. Although rule-based model proved as an efficient mechanism with result accuracy, it is suffering from the high amount of time and resource consumption, while executing the rule verification process. Another prominent insider threat detection model is "Activity based insider threat detection model". This will do monitor and verifies the total activities from all users to detect the insider involvement. As part of this detection model, each activity flow is compared against the pre-designed process representing graphs [6], which helps in finding the risk associated with an activity. The main limitation of this model is selection and execution of several verification policies at each level of the real activity.

Revised Manuscript Received on June 05, 2019

K. Venkateswara Rao, Ph.D Scholar at GITAM (Deemed to be University), Visakhapatnam, India and Associate Professor (CSE), Chebrolu Engineering College, Chebrolu, India.

Dr. T.UmaDevi, Associate Professor in the Department of Computer Science, GITAM (Deemed to be University), Visakhapatnam, India.

Another limitation is measuring the distance to classify the intended and un-intended insiders (to avoid false negative results), is a complex operation with result accuracy at very low. Recently another model become popular for insider threat detection is, Behavior based insider threat detection model [7], which identifies the malicious insiders by detecting the malevolent activities of them, using behavioral analysis theories. This theory mainly concentrates on employee personal traits and behavior patterns [8] of work environment. Although behavior based detection model recorded better performance than other detection models, it is also suffering from some major limitations are: i) identifying the malevolent activities become a multi-way diagnosis process ii) monitoring the personal traits is a time consuming and subject to privacy management iii) attained less accuracy in results due to false negatives. As part of protecting the clouds from malicious insider threats, Mr. K. Venkateswarao and Dr.T.UmaDevi et al [25] proposed an AI enabled Insider Threat Detection Model (ITDM) with respective abstract modules. The whole ITDM process module is implemented in this paper, with the state-of-the-art insider detection model as "Behavior Reliance Anomaly Detection (BRAD)", to find the malicious insiders precisely and to address the aforementioned insider threat detection limitations with cutting-edge technologies, in a comprehensive and detailed manner. The proposed BRAD is implemented as an integral part of Insider Threat Detection Model (ITDM), to detect the insider activities using log files data and design flows.

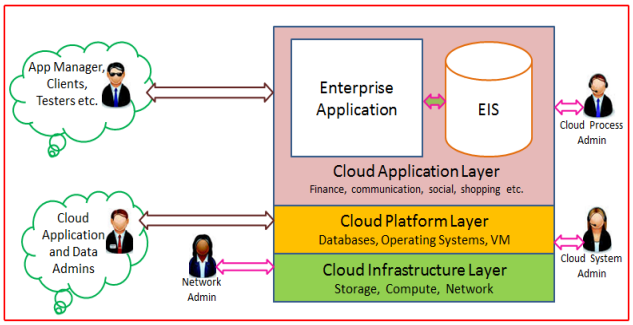

Fig-1. Cloud Deployment Model with Administrator roles

From the thorough analysis on malicious insider threat detection models [9 and 10] and their limitations, BRAD model concentrates on the less overhead, high speed, accuracy and the comprehensiveness as the main goals. To achieve these goals, BRAD designed with the custom novel techniques like CPT based Activity trees, Algebraic log query structures, four (unique, abnormal, suspicious and threat) layered verification process and Anomaly detection algorithm. In BRAD to authorize the insidious activity of an employee (admin),his activity relevant system logs and/or user activity logs have been verified against CPT base activity Trees. Using the CPT's, data access restrictions, past abnormal activities and CCIB, the BRAD classifies the admin activities as General, Unique, Abnormal, Suspicious and Threat. This four layered detection process helps in classifying the intended insider attacks and unintended (accidental) insider attacks precisely in a feasible and transparent manner. This layered model extensively helped in minimizing the complexity of threat detection process with fewer comparisons. It means BRAD is indirectly boosting the detection process speed and reducing the overhead on cloud by consuming very limited amount of resources for executing threat detection process.This BRAD is isolate and not required to involve into cloud personal code and activities, it is applicable and deployable to most of the cloud environments with minor cloud specific configuration changes. So the BRAD is portable and pluggable across different clouds. Anomaly detection algorithm validates the admin log data against CPT's, to determine the anomaly existence in admin activities. Comparison of BRAD against the other detection models with respective metrics has shown that, BRAD recorded the better performance than other detection models.Section - 2 presents the Literature review of malicious insider with case studies, Section - 3 describes the proposed Behavior Reliance Anomaly Detection (BRAD) approach in detail, Section-4represents the experimental analysis and result comparison process conducted on BRAD, and finally section-5 concludes this paper.

II. LITERATURE REVIEW

Cloud computing became a reliable computing source for enterprises and offering its services to organizations via SaaS, PaaS, IaaS etc. Low cost, high speed networks, on-demand resource requesting, pay-per-use money policy, high availability, reduced capital investments, remote connectivity and the other facilities are attracting the organizations to access and migrate their applications with the cloud. Apart from the cloud advantages, the other side organizations are much worrying about the cloud security threats especially the insider threats. So far many researches [4,6 and 8] have been concentrated to prevent the clouds from insider attacks, but they are suffering from several limitations due to the insider finding task is a sharp edge decision making process. In order to tackle the limitations in detecting malicious insiders from cloud environment, Mr. K.Venkateswarao and Dr.T.UmaDevi et al [25] proposed an "Insider Threat Detection Model (ITDM)" to identify the insider threats efficiently and to protect the cloud environment from them. ITDM is designed based on the standard malicious insider threat management flow is called "prevent-detect-respond" methodology. Each phase of this methodology is designed with respective components and algorithms. Prevent phase of insider threat management process is designed with the policies and guidelines. Most of these policies were designed by CSA as part of cloud security standards, which has to be followed by cloud services providers, while implementation of their clouds. The detect phase should be implemented technically, to deal with detection process of existed insiders and their activities. The final phase is incident response phase, in which the possible damage correction and unauthorized operation management is implemented. Among these three phases of ITDM, the detection phase is complex to design and implement as it is involved in detecting the malicious insiders based on their behavior with clouds. Hence the crucial detection phase of ITDM, is implemented with the help of our proposed Behavior Reliance Anomaly Detection (BRAD) approach. This paper explains about the BRAD theme, flow, modules, decision making and interconnectivity in section 3. Mahesh Babu et al [4], William R Claycomb et al [9] published papers on anomaly detection from insider activities revealed that the employee behavior and time line information plays a vital role in detection of malicious activities.

In support to this, the Ponemon Instituteconducted survey[2] revealed that, more than 84% of insider threats were taken 30 to 60 days of time period to complete their projected insider attack(series of operations). This information indicates that the execution of an insider threat is neither possible in a single day nor in a single attempt. Periodically the detailed verification of admin activities, helps in identify the insider threats in prior to damage and to prevent the cloud from insider threats.Log files (server logs, admin logs and process logs) are the simple text files, designed to maintain the constructive information about the executing activities of an application. Maintaining the log files is very common with each real life application, because they are very useful in tracing and debugging the application runtime exceptions. The detailed information about each activity and the sequence of application executed activities are recorded on log files at run-time. Hence Mahesh Babu et al [4] and F. L. Greitzer et al [8] used the available (system, application and security) log files to analyze the behavior of an employee (admin or any job role).All activities executed by admin/job roles are written to these files in a structural manner with detailed activity information like Emp_ID, Emp_Name, Job_Role, Begin_Time, End_Time, Act_Res etc. Eugene Schultz et al [24] designed framework explained the process of behavior analysis for insider threat detection using log data, verbal behavior and personal traits. A deep analysis has been conducted to know about the insider threats ensued in real life, and few of them are listed here as case studies. These case studies help to describe the nature of a malicious activity and guides how to detect the anomaly behavior of an insider.

***Case Study-I*:** Terry Child (a network admin)is responsible for California City's FiberWAN network management and monitoring. Due to his dissatisfaction on company, he suddenly became as malevolent and changed all admin passwords of California City's FiberWAN network [11] eventually, to get the full control on entire city's network. Once he executed the job of designing a centralized admin in successive set of actions, than he hijacked the entire city's network and refused to share the admin passwords even after the judiciary involvement. Due to this reason the California city's network had run for couple of days without network trouble shooting facility as the passwords hijacked by child. Finally he revealed the admin passwords in front of the judge and tried to support his activities by blaming the organization policies on employees. Perhaps he did not reveal the passwords, than the entire network system has to be reconfigured and users may lost the network access for couple of days.

Case Study-II: this case study explains about the procedure of how to catch the malicious activities in Payroll application. In payroll application the payroll admin is responsible to ensure the payroll activities done properly on time, and be able to access some payroll data fields (Emp_id, Emp_Name, Job_Role, E_Mail, Bank_Details, Loans, Allowances, Tax) of organization employees. To troubleshoot the issues of payroll department and to perform the essential activities on payroll database, a direct access and query execution interface will be provided to the payroll admin.In this case the payroll admin has a chance to access the other fields of employees (i.e. Job_History, Proj_Details etc.) or excess information than his job role boundaries (i.e. trying to calculate each project earnings and its share in organization's income) is considered as a malevolent activity. As part of his job, frequently the admin retrieves the employee's information (select f1, f2, f3 from employees of payroll_dept) from database within the access boundaries of his job role. To tamper the employee's information from database, payroll admin retrieves the employee data with some other fields (select all (*) from employees of payroll_dept) than allowed, which is the unique activity than regular. To accomplish the data tampering goal, the admin will tries to export the table data to system as an encrypted file and then starts sending this file (employee details) to his private email or USB devices. In this all employee data selection is a unique operation, exporting employee data from database to system is an abnormal operation, sending encrypted data file to a private email id is a suspicious behavior and finally it has to be declared as an insider threat.

Case Study-III: This case study explains about the UIDAI [23] developed Aadhaar number administration with different admin roles. This Aadhaar number is a proof of identity generated by Indian government to recognize the people uniquely. Most of the government and private agencies are using this Aadhaar number to avail their services to the people and to recognize their customers or beneficiaries exactly. Income tax department, financial organizations, Pension Schemes, Health care and property transactions are some example organizations taking Aadhaar support today. Recently the Supreme Court of India also strongly ordered that the privacy and security of Aadhaar should not be compromised at any level. For this big application management, several admins role have been created with different access restrictions. Aadhaar Owners, Enrollment admins, local update admins, Regional Head, access admins, Non-domestic enroll admins, application admins, data admins, cloud admins, system admins etc. Each admin role of UIDAI is having the specific access to a person aadhaar card fields. By using the UIDAI website [23], the owner can update his current address field of aadhaar card, but not allowed to update his name. Similarly UIDAI appointed local update admin can update the name, address, mobile number and email, but he cannot update the age or aadhaar number, which is possible with the regional admin only.To control the admin access on UIDAI data fields (columns), the design team provides access control structures with mnemonic expressions. These expressions explain about each admin accessible column name, type of access (CRUD) and access boundaries in detail, is considered as the Core Data Access Management (CDAM) base. Most of the real time applications use to implement this CDAM base, to specify the access restrictions of each admin on data fields. The design phase of SDLC uses the CDAM, to cross verify the out of bound activities granted to the admins. Section – 3 proposed BRAD model utilizes this CDAM base as activity base and transforms the mnemonic expressions to CPT's to detect the anomalies from admin activities.

Motivation: Literature review and the case studies on malicious insider attacks describing the facts are: Organization's negligence in implementing the prevention, detection and respond phases of insider threat detection is the main reason for this damage. Terry changed the all admin passwords of network environment eventually to manage the higher authorities without having

any doubt. From this incident our research noticed that, the Insider activity is not a single and simple activity to execute immediately, it is an activity which includes a series of operations to be performed gradually (i.e. explained in case study 2). Majority of the insider activities (based on their behavior) are anomalous, which means unique, abnormal and suspicious in nature. Henceforth there is a strong need of concentrating on designing the insider threat detection mechanism, to identify the malicious insiders precisely and to prevent the cloud from insider threats efficiently.

III. BEHAVIOR RELIANCE ANOMALY DETECTION (BRAD)

From the thorough analysis on insider activities, notified that "the malicious activity is a unique activity and is different from the regular activities of the admin". Malicious activity is not a single step operation rather it is a series of operations. Log files are the most popular resources (utilized by many former researches [4, 7 and 8]) to help in analyzing and detecting the malicious behavior of insiders. Insider threats are hard to detect and most dangerous than the other attacking mechanisms is explored by former researches [2 and 22].With the inspiration from past research works on protecting the clouds from insider attacks, this paper is introducing the detection phase of ITDM frame work [25] with Behavior Reliance Anomaly Detection (BRAD) as shown in figure2. The main goal of BRAD is detecting the malicious behavior of an insider by processing the system log files data. The proposed BRAD insider threat detection models classified into i) Validate Authentication, ii) Anomaly Detection, iii) Finding Abnormality, iv) Detecting Suspiciousness and v) Threat Declaration phases.In general, all activities of the cloud are sequentially recorded on log files as they executed in cloud. Before starting the real BRAD process, the main input resource log file data should be pre-processed, to support and speed up the BRAD process. To avoid the complexities in processing the log files data, this paper is suggesting that the log files and their data must be composed as per the NIST designed SP800-92 [12] log file management guidelines. These guidelines are most helpful in understanding and processing the log files automatically with log analytics. By following the NIST SP800-92 guidelines, our log files became the process feasible log records (data structures).Each record of the log file represents an activity done by the admin in detail. In this way all admins initiated activities are written to system logs for future reference. Admin_ID property of log record is a unique value, which is used to identify the activity initiated admin. As part of the log data pre-processing, all log records are clustered based on Admin_ID values (each cluster contains an admin related activities information). To identify the malicious activities from logs, each admin related log record set (cluster) is given as input to the BRAD main process.

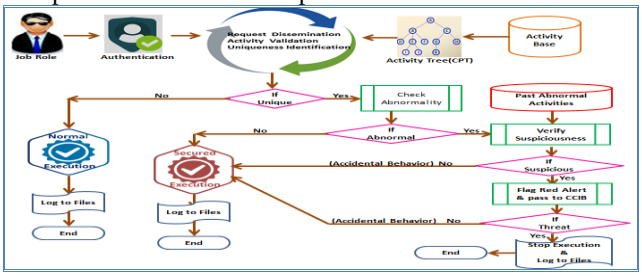

Fig-2 Design diagram of BRAD Model with Flow Patterns

A. Validate Authenticity

Upon receiving an admin related pre-processed log records, the validate authenticity module of BRAD verifies the logged in admin fellow is genuine or not. This module helps to detect the spoofing of identity committed by any cloud admin. Commonly the first few lines of user log record contain the login action related information like login event codes, login status and its description [13]. User_ID, IP_Address, Time Stamp, Login_Status, Error_Code, SSL certificates, DNS mapping, Server Tokens and Server info are the considerable attributes while authenticating the login of an admin. The process of validate authenticity is implemented as similar to Microsoft account logon audits [13] with some additional proprietary features. After completion of the admin login validation, the authenticated admin log records (admin cluster) will be sent to the anomaly detection module.

B. Anomaly Detection

The admin cluster is a collection of records and each record is representing an activity executed by the admin. This anomaly detection phase picks the admin records from cluster sequentially and validates each record to find the existence of anomaly. To detect the anomalies from user logs, the anomaly detection phase has to execute the three main processes are request dissemination, activity validation and uniqueness identification. As part of request dissemination the compact Prediction Trees(CPT) are generated from activity base (shown in figure 2.) and these trees are used to identify the anomalies from admin log records (i.e. admin activities).Here activity base is a custom data base, which consists of all possible system activities with execution flow information. The selected activities from the activity base are transformed to CPT's and these CPT's related process flow sequence is compared against the log recorded activity process flow sequence. Any deviation identified in this process flow sequence comparison indicates the chance of anomaly existence. In cloud, the admin initiated each activity is associated with an SQL query (i.e. retrieving the 5+ years experienced domestic employees from database). At the same time each admin is having the data access boundaries and limitations to preserve the privacy of cloud data. Generally an admin performs the CRUD operations (Creation, Retrieve, Update and Delete) on cloud application using SQL queries. Any attempt from the admin to access the unauthorized or illegal data by misusing their potential privileges leads to a malevolent activity. This anomaly detection phase main target is identifying such malevolent activities from admins by analyzing their behavior. Most of the times, a set of queries have to be combined and executed on database to accomplish the admin initiated activity. Each admin activity related system executed queries are written on log files sequentially. Apart from this the other additional activity related information like activity execution status, execution time and other effected dependencies also written on logs.As part of request dissemination process, an admin executed each activity related database queries are collected from logs and constructed them as a tree structure (as shown in fig-3). For example,

the payroll admin wants to select a set of employee records with salary summation value from department101to prepare the invoice report for management acceptance, before he pay the salaries as scheduled. For this activity, a query must be generated and executed on database to prepare a report of expected department employee's salary summation value. To detect the anomalies from admin activity relevant log queries they must be compared against the CPT's. But understanding and comparison (with CPT's) of these logged SQL queries (in text format) are not feasible for processing, hence these text queries need to be translated to relational algebraic expressions. Seri and Gottlob et al [14 and 15] proposed the SQL to algebraic translation guidelines in their research work. With the help of our custom dialects the algebraic statement symbols (**π, φ, γ, σ** etc.) are mapped to CPT based Activity tree operations (C, R, U, D, I, W etc.).These transformed algebraic expressions are constructed as query trees (figure3) to perform the comparison of them against the CPT's to identify the anomalies.

SELECT SUM (emp_sal) as "DEPT_SAL_SUM"FROM employees where dept_id=101 GROUP BY dept_id HAVING loan_due <=0

is translated to an algebraic statement as :

π *SUM (T.emp_sal) as "DEPT_SAL_SUM"* φ *T.loan_due* ≤ 0
γ *T.dept_id* σ *T.dept_id = 101* $\rho T(Employee)$

These algebraic structures are similar to DBMS memory executing SQL query logical forms, which are easily accessible and traversable. Using the translation process, all executed queries of each admin are transformed into the algebraic tree structures to compare against the CPT's for sake of anomaly detection. The SDLC [16]design phase of an application contain the "process sequence diagrams", which represents each application user activity with actors, relationships, flow sequence, data bases, decision makings, iterations etc. With the help of this process sequence and the other design models, the developer writes the application code using programming languages. At this level Core Data Access Management (CDAM) base is composed with admin related activities and data access restrictions in the form of mnemonic expressions (explained in case study - III).This CDAM base with access restrictions and activity flows is transformed as an activity base with minor changes. This activity base contains each admin related possible activities and the data access restrictions as shown in figure 2.

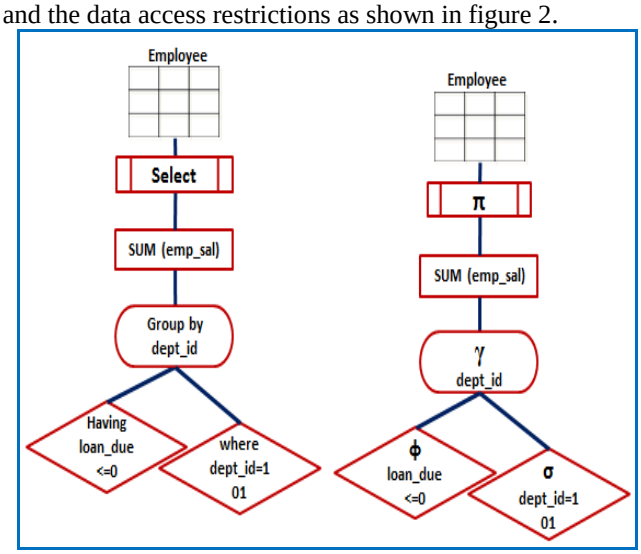

Fig-3. Translation of SQL Query to Relational Algebra

Now the CPT's are generated from activity base, to identify the anomalies from admin executed activities. Gueniche T et al specified that, the CPT's are the best sequence prediction models [17] and they are widely using in weather forecasting, stock market predictions, consumer product recommendations etc. BRAD utilizes these admin algebraic models to compare against the CPT's (blue print) to detect the anomalies. CPT's maintain the Inverted Index (II), Prediction Tree (PT) and Lookup Table(LT)at database each table level using the CDAM given information as shown in figure 4.CPT main components (II, PT and LT) are customized to support the proposed behavior analysis and anomalous detection process. In this customization, the Inverted Index consists of the admin access restrictions (CRUD) at each table column level, Lookup table consists of the list of possible sequences at each admin actor level, and finally the Predication Tree consists of inverted index table information. Figure 4 represents the construction of CPT for Doctor Table with associated columns and each column level access privileges (CRUD) to various admins like CA, PA and WSA.To illustrate the proposed CPT model, the Hospital Management System (HMS) application related Doctor Table is selected with id, name, salary, specialization, phone and certificate columns. Chief Admin(CA), Payroll Admin(PA) and Work Schedule Admin(WSA) are some admin actors of HMS, is authorized to manipulate the Doctor table columns using different SQL operations are create(C), Delete (D), update(U) and read(R) etc. Inverted Index table of figure -4, shows the relations among admins and table columns, which are maintained as sequences by Lookup Table (LT). For instance the below sequence S1_CA_DOC represents the Chief Admin data access restrictions at Doctor table each column level.

S1_CA_DOC = [CA, { (id→R,C), (name→R,C,U), (spec→R,C,U), (sal→R,C), (phone→R,C), (cert → R,C,U)}]

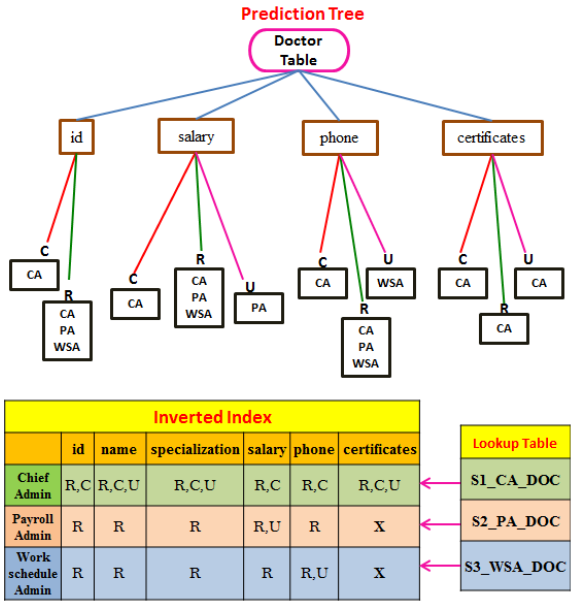

Fig-4 CPT based Activity Tree Representation for Admin Activities on Doctor Table

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

Figure-4 presenting inverted index table specifying that, the doctor's phone number column value is created by CA (HR) at the time of joining, but it can be updated only by WSA. Each Doctor related education certificates are created, read and updated by CA only, PA and WSA are even not allowed (X) to read them. Similarly the salary column value allows the CA to create and read, PA to read and update, WSA to read-only. In the same way as explained with Doctor Table, the entire Core Data Access Management Sequences (CDAMS) will be generated from cloud data and stored in sequence clusters for future operations. Sequences are created once in the beginning and they will be updated when the design flow is updated. To avoid the time and space complexity while designing the CPT based activity trees, this research is following the state-of-the-art compact prediction tree models [18]. These models are proved that, they are lossless, accurate, incremental, less in size, fast in prediction and recorded the low time complexity.Till now this section has described, how to transform the logged queries to process feasible algebraic statements and activity base to CPT's. Now it's time to start the validation of admin executed activities (algebraic statements) against the respective activity flows (CPT's) to identify the anomaly behavior of the admin. The below Anomaly Detection algorithm explains how to verify the user executed queries (algebraic statements) against the respective CPT's in detail.

C. Anomaly Detection Algorithm

Input: Activity base CPT's with PT,LT and II;User Activity Set (UAS), SVM classifier, custom dialects
Output:rs: predicted anomalies set;

```
Begin
user_activity : ua ;
anomaly = false;
resultSet rs;
Train_SVM (CPT's); // updating SVM knowledge with CPT's
foreach (ua∈UAS) do
anomaly := findAnomalyInSequenceWithSVM(ua){
                algebraic_seqas := SequenceDialect (ua);
                LT_seqlts := mapTemp_Seq(as , PT);
              InvertedIndexSet iis := getInvertedIndex(lts);
                return validateWithSVM(as, iis);
                    }
    If(anomaly == true) than
    rs.add(ua);
    end // if
  ua = ua.next();
end // foreach
return rs;
End
```

Anomaly detection algorithm initially trains the SVM classifier [19] with the Compact Prediction Trees (CPT's) and Core Data Access Management (CDAM) structures. In future this training knowledge helps the SVM classifier to identify the anomalies from given user activities (ua).To find the anomalies, admin executed all activities from log files are formed as a set (UAS) and each activity (ua) from this set will be given as input to *findAnomalyInSequenceWithSVM()* function to detect the anomalies. Next the SequenceDialect() function transforms the given user activity (log query) to algebraic structures as shown in figure 3. Using the mapTemp_Seq (ts, PT) function, the algebraic Temp Sequence (ts) is mapped with Prediction Tree (PT) to identify the relevant Lookup Table Sequence (lts). These LT Sequences are pointing to the corresponding rows of inverted index table. This lts information is given as input to get Inverted Index(lts) function to extract the relevant Inverted Index Set(IIS) information. This index information set (IIS) provides that sequence related access restrictions on different data fields of a table as shown in figure 4. Now this IIS information is used as a standard flow and the algebraic structure (ua) internal flow is cross checked against the IIS using validateWithSVM(as, iis) function. Here the CPT's trained SVM classifier compares the both input flows, and any variance notified in this comparison triggers the anomaly (unique) behavior in algebraic sequence (as). Likewise an admin executed all queries are verified using anomaly detection algorithm, later the identified anomalies are returned as a result set and the same information is written to BRAD result log files. This is the first level of the proposed malicious insider detection four layered process as shown in figure 2. The anomalies (resultSet) from this level are given as an input to the next level (abnormality verification) of insider threat detection layer, to verify the abnormality occurrence. Abnormal activity means the activity which is newly executed or rarely executed or uniquely executed or failed in the middle etc. As we knew that, the all unique activates are not insider threats, they may be executed accidentally or newly introduced to the job role. This second level insider threat detection layer compares the anomaly (unique) activities of first layer with their colleague admins activities and also verifies that query execution probability with admin to confirm the abnormality. If the result of this layer is normal activity, then it's left over execution details are verified to confirm this activity executed securely till end. In case the result of this layer is an abnormal activity, then that activity is given as input to the next level of insider threat detection layer (suspicious verification)to confirm whether it is suspicious or not. At this level, the abnormal input activity is verified against the previously recognized suspicious activities. If the selected abnormal activity found in the former suspicious activities list than it is mostly confirmed as malicious activity by insider, and the details will sent to the CCIB team for professional verification to declare as threat. The suspiciousness verification with recognized past malicious threats and the final CCIB verification process helps in overcoming the false negatives effect on result accuracy, which indirectly helped in improving the true negatives. Once the CCIB confirmed the executed activity as malicious insider activity, than the activity information will be written to BRAD logs and added to the past abnormal activities for future reference.

IV. EXPERIMENTAL RESULTS

In order to analyze the BRAD process and results accuracy, a prototype of the proposed BRAD approach is designed with respective technologies. This prototype is deployed with the hospital management software application, to monitor and detect the malicious activities in that approach. As aforementioned, my research designed the BRAD prototype as a portable plug-in app to support the

insider threat detection from all applications with minor configuration changes. BRAD prototype expects the log files and design flow as input and periodically monitors (online/offline) the admin or any job role activates to report the malicious behavior. This prototype has been deployed along with the application infrastructure and executed for a certain amount of time period. To test the BRAD monitoring capabilities some temp admits are created and started executing a set of predefined activities (normal and abnormal both) with sample data on hospital management system. The below table-1representsthe BRAD prototype mock admin roles, each admin relevant sequences from inverted index table, executed test activities(test cases) and results count of BRAD four (unique, abnormal, suspicious, threat) layers.

Mock Admins	Seq Count	Executed Activities	Unique Count	Abnormal Count	Suspicious Count	Threat Count
Chief Admin_x	6,193	323	82	19	8	7
Payroll Admin_x	2087	304	39	8	1	1
Work Schedule Admin _x	1238	427	71	13	6	4
Medical Store Admin_x	2845	284	62	26	10	8

Table1. BRAD Prototype experimental statistics

In order to evaluate the efficiency of the proposed BRAD approach, my research compared the results of BRAD against the other behavior based anomaly detection techniques are Static Anomaly Detection (SAD) [20], and Specification based Execution Monitoring [SEM] [21]. These three former approaches were designed to identify the malicious activities based on behavior analysis while executing the activities of respective software application. Static Anomaly Detection (SAD) analyzes only the security anomalies based on design flow models, whereas our BRAD detects the different types of admin behavior models like normal, anomaly (unique), abnormal, suspicious, threat etc. SAD designed call graph model just verifies the execution flow order and any unexpected result or termination returned as anomalous. Our BRAD verifies the log records against CPT design sequences and validates the anomaly behavior at four different levels (unique, abnormal, suspicious and threat) to separate the accidental (unintended) anomalies. Calvin at el designed the Specification based Execution Monitoring [SEM],is a formal intrusion-detection model to identify the malicious insiders using traces and order sequences. SEM proposed formal language maintains a program related execution process order as sequence. This sequence is considered as a specification and any object executed operation flow will be compared against sequences to determine the insidious activiSEM proposed sequences and trace policies are appointed only to check the order of execution to identify the intended behavior, but our BRAD is designed the sequences to maintain the access flow and data access permissions (CDAM) to detect the insider activities comprehensively at process level and data access level. SAD and SEM frameworks classifies the given activity is malicious or not, but proposed BRAD four layered mechanism classifies the input activity as normal, anomaly, abnormal, suspicious, threat, accidental or intended threat. This detailed classification from BRAD helps to address the anomalies in an efficient manner. Our BRAD utilizes the recorded log data and application design content to analyze the admin user behavior without interfering to application code and execution. Proposed BRAD is configured to run the threat detection process, at specific time intervals when the server resources are available without disturbing the main services of server. The ITDM framework mostly runs the BRAD process at server idle time, so BRAD cannot negatively impact on server's performance and load. At the end BRAD recommends the suspicious activities list to the CCIB officer's verification for final confirmation of them to declare as insider threats. This verification is used to remove the false negatives from results and it helps to improve the result accuracy.

V. CONCLUSION

As part of my research on protecting the clouds from Malicious Insiders, the Insider Threat Detection Model (ITDM) has been proposed with the standard threat mitigation modules are prevent, detect and response. Among the modules of ITDM, the complex insider threat detection module is comprehensively designed in this paper as BRAD to identify the malicious insider activities. This paper describes the design and workflow of "Behavior Reliance Anomaly Detection (BRAD)", along with its four level threat detection process. In BRAD the CPT's are considered as specification and the admin activity relevant algebraic statements are compared against CPT's to identify the malicious activities. Anomaly detection algorithm played a vital role in detection of anomalies from user activities. The four layered mechanism of BRAD is facilitated to classify the intended and unintended malicious insider activities clearly. Along with the process design sequences, the data access permissions (CDAM) also considered while validating the user activities. BRAD prototype based experimental analysis specified that, our BRAD is proved portable, reliable, scalable and comprehensive when compared with the former behavior based anomaly detection methods.

REFERENCES

1. Cloud Security Alliance "Top Threats to Cloud Computing : DeepDive" https://cloudsecurityalliance.org/artifacts/top-threats-to-cloud-computing-deep-dive//
2. Ponemon Institute "2018 Cost of Insider Threats: Global Organizations"https://www.observeit.com/ponemon-report-cost-of-insider-threats/
3. Kandias, Miltiadis, Virvilis, Nikos, Gritzalis and Dimitris. "The Insider Threat in Cloud Computing". International conference on Critical Information Infrastructure Security, Springer Berlin Heidelberg, pp 93-103, 2013.
4. B. Mahesh Babu and Mary Saira Bhanu "Prevention of Insider Attacks by Integrating Behavior Analysis with Risk based Access Control Model to Protect Cloud" Eleventh International Multi-Conference on Information Processing (IMCIP-2015), Science Direct Procedia Computer Science, pp 157 – 166, 2015.
5. Andrew P. Moore, Dawn M. Cappelli and Randall F. Trzeciak "The Big Picture of Insider IT Sabotage Across U.S. Critical Infrastructure". CERT Technical Report, CMU/SEI-2008-TR-009, May 2008.https://resources.sei.cmu.edu/asset_files/TechnicalReport/2008_005_001_14981.pdf
6. Naghmouchi, N. Perrot, N. Kheir, R. Mahjoub, and J. Wary, "A New Risk Assessment Framework Using Graph Theory for Complex ICT Systems". In Int. Workshop on Managing Insider Security Threats. ACM, Vienna, Austria, pp 97–100 October 28, 2016.
7. S. Aurigemma and R. Panko, "A Composite Framework for Behavioral Compliance with Information Security Policies," in 45th Hawaii International Conference on System Sciences,Maui, HI, USA, pp. 3248–3257, 2012.

8. F. L. Greitzer, R. E. Hohimer, and A. Biography, "Modeling Human Behavior to Anticipate Insider Attacks",Journal of Strategic Security : Summer 2011- Strategic Security in the Cyber Age, vol. 4, no. 2, pp. 25–48, 2011.
9. W. R. Claycomb and A. Nicoll, "Insider Threats to Cloud Computing: Directions for New Research Challenges," 2012 IEEE 36th Annual Computer Software and Applications Conference, Izmir, pp. 387-394, 2012.
10. Ivan Homoliak, Flavio Toffalini, Juan Guarnizo, Yuval Elovici, Martín Ochoa "Insight into Insiders and IT: A Survey of Insider Threat Taxonomies, Analysis, Modeling, and Counter measures"ACM Computing Surveys (CSUR). Vol. 52 (2019), Issue 2, e-print arXiv:1805.01612, May 2018.
11. Robert McMillan, "Terry Childs gets four year sentence" AUGUST-06,2010. https://www.csoonline.com/article/2136897/core-java/terry-childs-gets-four-year-sentence.html
12. Karen Kent and Murugiah P. Souppaya. "NIST SP 800-92. Guide to computer security log management". Technical report, Gaithersburg, MD, United States, September-2006.
13. Justinha and Andrea Bichsel"Audit logon events" April-2014. https://docs.microsoft.com/en-us/windows/security/threat-protection/auditing/basic-audit-logon-events
14. Alexey Grigorev "Translating SQL to Relational Algebra", 25-December-2013. http://mlwiki.org/index.php/Translating_SQL_to_Relational_Algebra
15. S. Ceri and G. Gottlob, "Translating SQL Into Relational Algebra: Optimization, Semantics, and Equivalence of SQL Queries," in IEEE Transactions on Software Engineering, vol. SE-11, no. 4, pp. 324-345, April 1985.
16. Shadi A.Aljawarneha, AliAlawnehb and ReemJaradatc "Cloud security engineering: Early stages of SDLC" Future Generation Computer Systems, Volume 74, Pages 385-392, September 2017.
17. Gueniche T., Fournier-Viger P., Tseng V.S. "Compact Prediction Tree: A Lossless Model for Accurate Sequence Prediction". Advanced Data Mining and Applications (ADMA), vol 8347. Springer, Berlin, Heidelberg, PP-177-188, 2013.
18. Gueniche T, Fournier-Viger P, Raman R and Tseng V.S "CPT+: Decreasing the Time/Space Complexity of the Compact Prediction Tree". Advances in Knowledge Discovery and Data Mining in Pacific-Asia Conference on Knowledge Discovery and Data Mining(PAKDD), vol 9078. Springer, 2015.
19. Gjorgji Madzarov, Dejan Gjorgjevikj and Ivan Chorbev "A Multi-class SVM Classifier Utilizing Binary Decision Tree" published in Informatica vol-33,PP 225-233, 2009.
20. D. Wagner and R. Dean, "Intrusion detection via static analysis," Proceedings 2001 IEEE Symposium on Security and Privacy. Oakland, CA, USA, pp. 156-168, 2001.
21. C. Ko, M. Ruschitzka and K. Levitt, "Execution monitoring of security-critical programs in distributed systems: a specification-based approach," Proceedings. 1997 IEEE Symposium on Security and Privacy (Cat. No.97CB36097), Oakland, CA, USA, PP 175-187, 1997.
22. Team ObserveIT "Five Examples of Insider Threat-Caused Breaches That Illustrate the Scope of the Problem"March-22,2018. https://www.observeit.com/blog/5-examples-of-insider-threat-caused-breaches/
23. Unique Identification Authority of India "Aadhaar Authentication Api Specification - Version 2.0,February-2017" https://uidai.gov.in/images/FrontPageUpdates/aadhaar_authentication_api_2_0.pdf
24. E.Eugene Schultz "A framework for understanding and predicting insider attacks" Journal of Computers and Security, Volume 21 Issue 6,PP 526-531,Elsevier Advanced Technology Publications Oxford, UK October-2002.
25. Mr.K.Venkateswara Rao, Dr. T.Uma Devi "Architecture of Insider Threat Detection Model to Counter the Malicious Insider Threats on Cloud" JASC: Journal of Applied Science and Computations, Volume 5, Issue 10, PP 1360-1368, October-2018.

AUTHORS PROFILE

K.Venkateswara Rao pursing Ph.D. from GITAM University, Visakhapatnam and also working as Associate Professor in the Department of Computer Science, Chebrolu Engineering College, Chebrolu, A.P, India. His interested research domains are Cloud Security, Machine Learning, Green Software design and Artificial Intelligence.

Dr. T. Uma Devi is working as Associate Professor in the Department of Computer Science, GITAM (Deemed to be University), Visakhapatnam, India. She did her PhD from Nagarjuna University in 2011. Her research interests include Neural Networks, Data Mining, Bio Informatics, Data Analytics. She has been published good number of research publications in reputed international journals and guiding good number of research scholars for Ph.D.