International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 4, April - 2013

Uncertain Event Processing Using Prediction Correction Paradigm

V. Govindasamy[1]
Pondicherry Engineering College, Pondicherry, India
P. Thambidurai
Perunthalaivar Kamarajar Institute of Engineering and Technology (PKIET), Karaikal,Pondicherry,India

Abstract–The systems which react automatically to events must be capable of handling abundant load of incoming events the problem resides in the designing of the system is handling events associated with uncertainty and materializing the events by the occurrence of the relevant events at active functionality in higher rate of accuracy. Wide spread paradigm for such materializing is Complex Event Processing, a rule based paradigm, the rule definition is purely depended on the domain experts, the domain experts have to provide the necessary event for the processing and those rules will be processing the incoming events and pass out the input to the system. While it is reasonable to expect that domain experts will be able to provide partial rules specification, providing all the required details is a hard task, even for domain experts [2]. Moreover, in many active systems, rules may change over time, due to the dynamic nature of the domain. Such changes complicate even further the specification task, as the expert must constantly update the rules. To lower the burden of the domain expert to constantly update the rule we device this mechanism of both defining of rules at the initial stage and update the rule over the time. It combines both the information provided by the domain expert with machine learning technique it is aimed at improving the accuracy of event specification and materialization

Index terms-Complex event processing, rule-based reasoning with uncertain information, prediction correction paradigm.

1. INTRODUCTION

The complex event processing has gained interest many area of engineering scientific and public beneficiary security and so on applications all these areas of application requires sophisticated mechanism for manage events and materialize new events from the existing ones, event materialization involves process like *sampling* and *selectability* these are represented using rules[1]. The rules that has to be implemented for the selecting and sampling of the events must be implemented by the domain experts which is a tedious work, once the rule implemented will not remain constant since the system has an changing nature of Operation, so the rules must also be changed according to the changes occur in the system and the problem that is involved in This process is identifying the parameter for the rules and implementing the rules as soon as possible in order to avoid the loss in data

2. PROBLEM DEFINITION

The problems concerned with uncertainty event derivation are, In many cases such derivation is carried out based on a set of rules [7]. The uncertainty event derivation is made to be in a ruined position due to the in ability to relate the actual occurrences of events continuously, to which the system must respond, and the ability of event-driven systems to accurately generate events. This results in uncertainty and may be caused by the defective event sources. The next challenging task is to determine with confidence whether the event has occurred or not with the provided source of information is highly recommended for the need to enable timely response to events [6]. Event derivation should also scale for a large number of loads of event. Assigning rules for processing events under large scale of events input from the sources and processing all the events with the rules to derive the new events will not be an efficient way of derivation. This will cause a gap in the derivation of events and materialization of events. To avoid this back log an effective frame work is been utilized for event derivation and the rule associated with the processing of events has to be implemented by the domain experts, such an monotonous work cannot be carried out efficiently. In order to avoid the inefficient implementation of the rules, and unproductive derivation of the events once the rule implemented can be tuned by machine learning techniques [2] and can be updated according to the change of course of the system. We had devised an intelligent frame work that will efficiently process the uncertain events and tune the rule parameter assigned to process the event

3. RELATED WORK

This frame work gets uncertain events as inputs from the systems connected to the network that is processed over a set of rules contained in the server to determine that whether the event is eligible for derivation of other events or not. The event derivation is based on the set of events that are predicted for the derivation after processing through the set of rules the newly derived event's probability is computed through the set of probability spaces defined over the time *t* since the probability space of the derived event varies for time to time. The sampling function is implemented to generate a Bayesian network that is used to predict a probability of the events efficiently and approximate the occurrence of the event and selectability technique is used in the rule parameter to select which events are eligible for the derivation of the events. The rule implied in

the selectability technique often tends to change first of all the implementation of those rules itself a painstaking task for the domain experts. The frame work we suggest will be tuning the rule parameters defined and update the rule parameter in the due course of changes in the system this is achieved by using machine learning technique (*a simple statistical learning technique*) implemented through a *prediction - correction* paradigm the *rule prediction* stage tunes the rule by utilizing the intelligence gathered through the process of learning about the rule without any human assistance and the *rule correction* process will change the tuned rule with the intelligence provided by the domain expert.

3.1 EVENT COMPOSITION

Various systems enabling event composition have been defined. Some of these are designed specifically for active databases. Others are general-purpose event composition languages. All existing languages enable deterministic inference of events, based on a set of rules. Each rule describes a complex sequential predicate or function, based on which inference is carried out .A major shortcoming of all existing specification languages is that they can reason only about deterministic knowledge regarding the relations between events. In addition, none of the existing systems are designed to handle uncertainty in a general and formal manner.

3.2 MECHANISMS FOR PROBABLISTIC REASONING

The commonly used approach for quantifying probabilities is Bayesian networks. However, Bayesian networks are only adequate for representing true or false probabilistic relationships between units. In addition, standard Bayesian networks cannot clearly model sequential relationships. To overcome these limitations, several extensions to Bayesian networks have been defined, including Dynamic Belief Networks, Time Nets, Modifiable Sequential Belief Networks, and Sequential Nodes Bayesian Networks. Although these extensions are more expressive than classical Bayesian networks, they nonetheless lack the expressive power of First-order logic. In addition, some of these extensions allow more expressive power at the expense of efficient calculation. Another formal approach to reasoning about probabilities involves probabilistic logics. These enable assigning probabilities to statements in first-order logic, as well as inferring new statements based on some axiomatic system. However, they are less suitable as mechanisms for the calculation of probabilities in a given probability space. This approach combines the representational strength of probabilistic logics with the computational advantages of Bayesian networks. In this paradigm, separate models exist for probabilistic knowledge specification and probabilistic inference – i.e., probabilistic knowledge is represented in some knowledge model, whenever an inference must be carried out, an inference model is constructed based on this knowledge. In this work, we have chosen an approach very similar to this paradigm: Knowledge is represented as probabilistic rules, while probability calculation is carried out by constructing a Bayesian network based inference model.

3.3 UNCERTAINTY WITH EVENTS

Uncertainty is defined as the lack of certainty a state having limited knowledge where it is impossible to exactly define the existing state a future outcome or more than one possible outcome. The uncertainty in event processing system is caused by unreliable source or faulty source of information these results in the gap between the actual occurrences of explicit events. The above situation is the reason for the hampering of event derivation by the system based on the gap or lack in tracking input events. Two main aspects that must be considered while developing an uncertain event processing system are derivation should scale under heavy loads of input events and the probabilities associated with the derived events must be captured and correctly represented. One of the best ways to handle

3.4 INTRUSION DETECTION

[11] Describes a general model of IDS. The information system being protected (application, computer and/or network) is subject to a usage configuration or policy that describes legitimate actions of each entity (user, host or service) profile. Audit data describing entity actions or system states are generated (either systematically or triggered by the IDS) and then analyzed by the IDS, which evaluates the probability of these states or actions being related to an intrusion. Data processed by the IDS may be a sequence of commands executed by a user, a sequence of system calls launched by an application (for example a Web client, network packets, etc.) Finally, the IDS can trigger some countermeasures to eliminate attack cause/effect, whenever an intrusion is detected. Intrusion detection technique is separated into *signature-based* and *anomaly-based detection*. The former must possess an attack description that can be matched to sensed attack manifestations. This can be as simple as a specific pattern that matches a portion of a network packet or as complex as a state machine or neural network description that maps multiple sensor outputs to idea attack representations. If an appropriate idea can be found, signature-based systems can identify attacks that are abstractly equivalent to known patterns. Signatures that are too specific will miss minor variations on a known attack, while those that are too general can generate large numbers of false alarms. Such systems are inherently unable to detect truly novel attacks and suffer from a high rate of false alarms when signatures match both intrusive and non-intrusive sensor outputs. Anomaly-based detection equate *unusual* or *abnormal* with intrusions. Given a complete characterization of a noise distribution, an anomaly detector recognizes as intrusion an observation that does not appear to be invalid data alone. Such a mechanism needs to be trained on those unwanted data and has difficulty tracking natural changes in the noise distribution. These changes can cause false alarms while intrusive activities that appear to be normal may cause missed detection. Signature - based detection is more natural for rule representation, its sensitivity to changes helps to seek a rule representation for anomaly-based detection, a more challenging task. In this work we define rules based on the probability distribution function of explicit events features. By that, we decrease the role of domain experts in the rules formulation process to recognition of impacting factors, letting the system self-tune rule parameters.

4. MODEL

An event is actually an occurrence or happening that is significant (falls within a domain of discourse)and atomic (either may occur or not).based on the idea given in [1] We differentiate events two types they are *Explicit Events* these events are signaled by external sources or the events that occurs outside the system e.g. signal got from the sensors to a system, and *Implicit Events* these events doesn't have any direct signaling but they are yet to be derived by the system based on the events that are outside and related to the system. A discrete model of time is used in this system that has some time points that are used to reference the events occurrences. Let (t_1, t_2...) be a sequence of time point the high level of data contained in this discrete time intervals depend upon the application.

4.1 REPRESENTATION OF EVENTS

Events are represented in the system by associating the events with data; events share some common data types with them, data types like occurrence of time are common for all events but data type associated with the event are specific. *Attributes* is the name coined for the terms associated with the events. The event is recorded with the Event Instance Data (EID) Events with uncertainty can be represented using a single tuple of values <val_1, val_2...val_n> but when an event is with uncertainty it should be represented with several tuples and one of those tuples must have the probability of occurrence the event, and other attributes of the events. There is a probability associated with the events it is the complementary probability which gives information that the event had occurred or not. Due to the close world representation of the events the event will be considered as not occurred unless it is represented explicitly by its Event Instance Data. An event history is denoted by ${}_{t1}h^{t2}$ represents the set of all events as well as their associated data fall within the time t_1 to t_2. And the system event history is the set of all the events occurred with the Event Instance Data which has occurred in the entire system it is denoted by ${}_{t1}H^{t2}$

4.2 RULES

Rules represent information on event relationships each rule serves as a template and can be applied at a given time t. to event histories that are known at time t. the result of applying a rule to a specific event history may serve for the inference of, at most a single event. A rule r is given in the form <sel_r^n, $pattern_r^n$,$eventType_r$, $mappingExpressions_r$, $prob_r$>
where,
sel_r^n is a deterministic predicate returning a subset of an event history of size less than or equal to n (for some integer n).
$pattern_r^n$ is a (possibly complex sequential) predicate of n over event instances (note that this is the same in appearing in sel_r^n
$eventType_r$ is the type of event inferred by this rule.
$mappingExpressions_r$ is a set of functions mapping the attribute values of the events that triggered this rule to the attribute values of the inferred event.

$prob_r \in [0,1]$ is the is the probability of inferring the event by the rule. The predicates defined by sel_r^n and $pattern_r^n$ are deterministic, as are the functions $mappingExpressions_r$ therefore, the only uncertainty present in the rule is represented by the quantity $prob_r$ Indeed, many deterministic composite event languages can be viewed as defining a set ofrules R such that each rule r∈R is of the form <sel_r^n, $pattern_r^n$, $eventType_r$, $mappingExpressions_r$, $prob_r$>

4.3 SYSTEM STATE AND FEEDBACK

It may seems easy to define the rule parameters by one to specify but it is difficult to define the exact parameters for the rule therefore the system is designed actually for the purpose of automating the process of parameter specification. The set of all rules is collectively defined to be the system state at the time interval if given a time interval $T_k = [t_i^k, \ldots t_j^k]$ and the n rule parameters at time interval $T_k$as X_k belongs to R^n the system the devised system updates the rule in two stages in two ways namely rule parameter prediction and rule parameter update in the former stage the rule parameters are updated without any domain expert supervision and it is based upon the previous EID's that are recorded before any changes in the rule and the also with the intelligence of how the parameters may change over the time as well as on the constantly updated history h of explicit and materialized events. In the later the updated rule is tuned by the domain expert supervision i.e. feed back by the domain experts the feedback is of two types they are *direct feedback* and *indirect feedback*Direct feedback involves changes to the system state while indirect feedback provides an assessment on the correctness of the estimated event history *h* for example an direct feedback may be actual minimal packet length indicating an network attack inferred by the system and indirect feedback may be making an event e in h as non exist ant or by suggesting an event occurrence at time t so that e is not in ***h***

5. RULE TUNING MECHANISM

Rule Tuning mechanism involves two stages they are rule parameter prediction and rule parameter correction the forms works based on the constantly updated history of materialized events and allow the inference of complex event using the predicted parameter values is based on expert feedback of actual occurrence of predicted events and the recently materialized events allowing update to rule parameter in preparation for the next prediction stage.

Rule	Rule Purpose	Input Events	Input events	Output events
$r_1$src_bytesRule	Variance in level of packet size from source to destination	E_0	Score=C*prob(E_1)*weight(E_1)	E_1 variance of src_bytes, Features = {score}
r_2 dst_bytesRule	Variance in level of size from destination to source	E_0	Score=C*prob(E_2)*weight(E_2)	E_2 variance of dst_bytes attributes = {score}
$r_3$serror_rateRule	Variance in level of connections that have	E_0	Score=C*prob(E_4)*weight(E_4)	E_3 variance of srv_ratettribute features = {score}

	"SYN" errors			
$r_4$srvrateRule	Variance in percentage of service to different hosts	E_0	Score=C*prob(E_5)*weight(E_5)	E_4 variance in srv_rate attribute features = {score}
r_5 if attackRule	Attack detection rule	E_0, E_1,E_2,E_3,E_4	According to algorithm 1	E_5 attack, features = {score, logged_in, if_attack}

Table 1: inferred events and rule logic

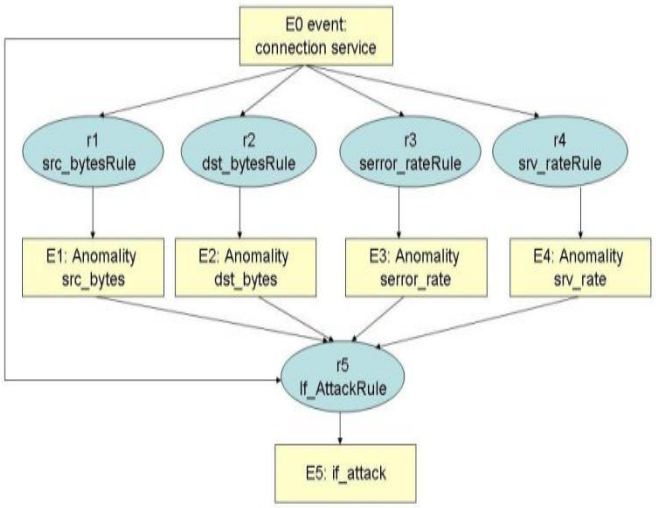


Figure 1: a rule tree example

Example1. An Intrusion Detection System (IDS) receives explicit events about new service connections, recognizing which of these connections are intrusion attempts. The inference mechanism of this IDS is defined as a rule tree, as illustrated in Figure 1 [2]. Rectangles represent events and ovals represent rules. Each of the rules r_1, r_2, r_3, $r_4$is aimed at, first, analyzing a feature of an explicit event and, second ,inferring a new feature anomaly event defining the extent to which this connection session attribute is considered to be anomalous. The inference logic of these rules is as follows:

- Rule r_1, srcBytesRule, infers event of type E_1, representing the variance in level of the packet length received from the source, based on the learned normal distribution parameters.
- Rule r_2, dstBytesRule, is similar to r_1 and infers event of type E_2, representing the anomaly level of the packet length received from the destination user, based on the dst-bytes attribute of the explicit event E_0.
- rule r_3 , serrorRateRule, based on the serror Rate explicit event attribute, infers event of type E_3 , representing the anomaly level of percentage of connections within a 3-minute time window that have "SYN" errors.
- Rule r_4 , based on the srvRate explicit event attribute, infers event of type E_4 representing the anomaly level of percentage of different open connections within a 3minute time window.

Rule r_5 analyzes the inferred attribute anomaly events of types E_1, E_2, E_3E_4, inferring the integrated anomaly event(type E_5). The rule parameter, a threshold of an integrated anomaly level, is learned using Algorithm 1. Table 1 defines rule parameters for each explicit event.

Algorithm 1 infrence algorithm for rule r_5

1. *if attack* ← *false*
2. *total score* ← *calculateScoresSum(E_1,E_2,E_3,E_4)*
3. **if** *(loggedIn* ∧ *totalScore > totalScore1)* **then**
4. *if attack* ← *true*
5. **end if**
6. **if** *(¬loggedIn* ∧ *totalScore > totalScore2)* **then**
7. *if attack* ← *true*
8. **end if**
9. **return** *if Attack*

5.1 PREDICTION CORRECTION PARADIGM

The prediction process studies the rule implemented in the system and it tunes the rule based upon the history of events derived based on the recorded Event Instance Data. The prediction process involves the statistical estimator (kalman filter) that learns about the rule and the machine learning technique tunes the rule without any domain expert supervision The correction paradigm is used to verify the tuned rule with the intelligence provided by the domain expert and the rule will be tune according to the feedback again. The entire process is based upon the parameter defined by the domain expert to tune and update the rule.

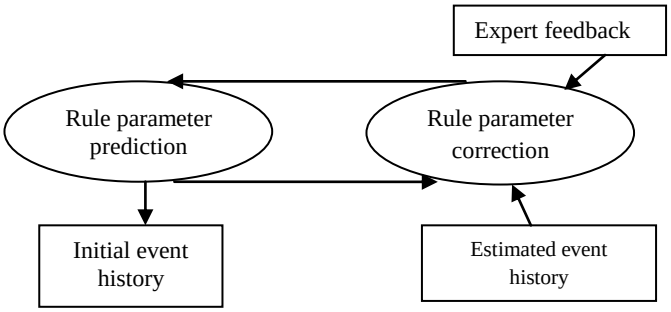


Figure 2: Prediction Correction Paradigm

The prediction correction process working mechanism is been described in figure 2 this paper make the following contribution

- We imply an simple yet powerful mechanism to efficiently process uncertain events in rule based system
- We present a mechanism to implement the parameters to the Rules in the rule based system automatically with an intelligent technique, combining knowledge possessed by the domain expert.
- A simple model is presented for rule parameter determination and updating with the combination of

both existing knowledge regarding the updates of rule parameters and indirect expert feedback.

6. SELECTABILITY

Selectability, is denoted by function S_r in a rule specification, plays an important role in event derivation, in both the deterministic and the uncertain settings. First, it defines which events are relevant to derivation according to rule *r*-an important semantic distinction. Just by analyzing the definition of Sr it is clear to a human which events are defined as being relevant to derivation according to r, and which events are ignored in this derivation. Selectability significantly influences the performance of the inference algorithm. In all tests, the performance deteriorates by orders of magnitude with the increase of relevant events. This, therefore, demonstrates the important role played by selectability in ensuring efficient processing of events. To explain, note that without the ability to filter out events that are irrelevant to derivation event derivation would always be carried out in a setting similar to the complex setting, in which most (or even all) events would be considered relevant to derivation. The details of selectability is been defined in [1].

7. SAMPLING

The sampling algorithm generates a Bayesian network [5] from which the exact probability of each can be computed given an existing Bayesian network, it is also efficiently possible to approximate the probability of an event occurrence using a sampling algorithm the mechanism of probabilistic reasoning is achieved through the sampling procedure The algorithm RuleSamp[1].

7.1 PROBABILITY SPACE DEFINITION

Probability space representation must be capable of representing the data associated with the events. the probability space representation is carried out through the representation of events that occurred at time *t* by representing all the events occurred at time *t* with a set of events called as worlds *W* the world contains the Event Instance Data (EID)[7],[3] of the occurred events at time *t*, a finite number of event representation at a possible world will have a finite number of EID's and each of them will be assigned with a possible number of *tuples* these *tuples* represents the EID's in world the probability space at time *t* can be represented using a triple $p_t = (W_t , F_t , \mu_t)$,where W_t represents the set of possible worlds, with each possible worlds related to a specific possible event history at time *t*, F_t is an algebraic function $F_t \subseteq 2^{|W|}$ over the set of worlds W_t and μ_t is a probability evaluation function over F_t where $\mu_t : F_t \rightarrow [0,1]$ we assume that that the event history associated with each possible world is finite and real world is one among those possible worlds, consequently each world is a finite event history consists a finite number of EID's.

8. STATISTICAL ESTIMATORS

The mechanism of rule tuning model devised in this frame work has the following functionalities estimation of the current rule state and updating the rule, later with the partial information provided by the domain expert. For the implementation of such strategy a machine learning technique is needed the family of statistical optimal estimators satisfy these properties, it is a simple computational algorithm that process the measurements to deduce the minimum error estimate of the state of a system by utilizing knowledge of system and measurement dynamics statistical estimation involved with three main type of problems namely filtering smoothing and prediction filtering is defined as the time at which an estimate coincides with the last measurement point,
Smoothing is defined as the time of interest falls within the span of available measurement data, prediction is defined as the time of interest occurs after the last available measurement,
This frame work is based on the kalman estimators which is a simple type of supervised Bayesian predict estimators and therefore preserve all the desired properties of the machine learning model for our needs.

8.1 KALMAN FILTER

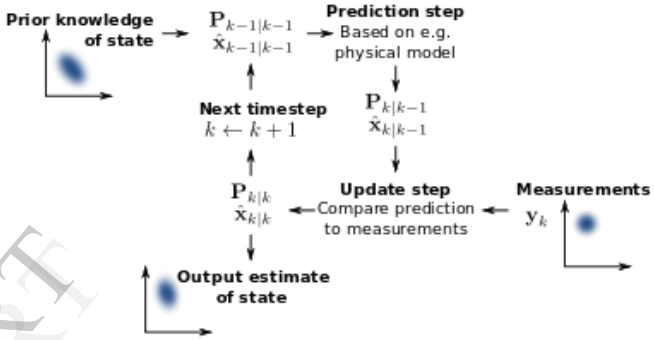


Figure 3: kalman filter

Figure 3 depicts the working of kalman filter the working can be described as the filter keep tracks of the estimated state of the system variance and uncertainty of the estimate. The estimate is updated using a state transitional model and measurements. $\hat{x}_{k|k-1}$ represents the estimate of the system state at time step *k* before the *k* measurement y_k has been taken into account $P_{k|k-1}$ is the corresponding uncertainty. The Discrete kalman filter equations [12] can only be used in the presence of direct expert feedback i.e. when providing revisions to the parameter values only.

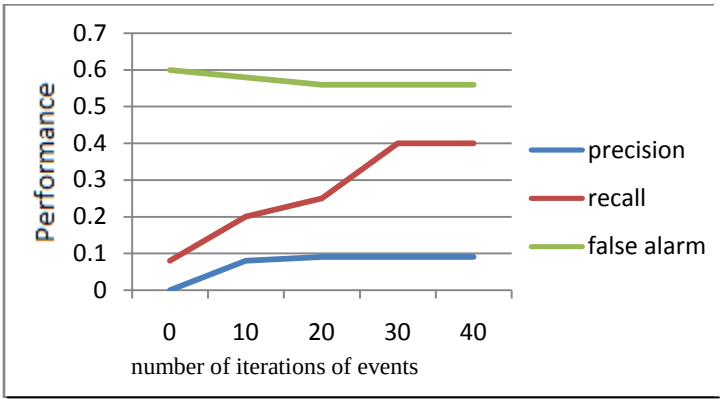
9. DATA SET AND EXPERIMENTAL SETUP

To measure the efficiency of the system the possible parameters that were used are possible number of worlds constructed, number of events eligible for derivation, efficiency of the statistical method, the experimental setup had a performance prediction method by the formula of ratio between number of events generated and total processing time the system was implemented in Java and an simulation run was conducted to get the average efficiency value of the system. A sub set KDD'99 of DARPA off-line intrusions detections evolutions data sets was taken as the model set up and implemented.

9.1 EXPERIMENTS CONDUCTED AND THE RESULTS

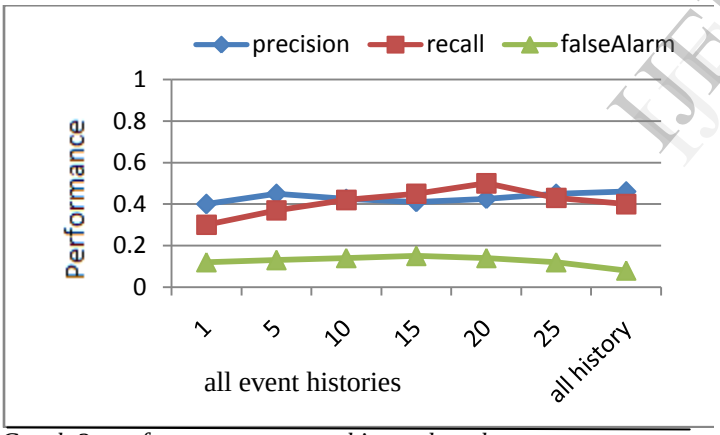
The study of learning probability of the model is experimented the two parameters that where aimed at the presence were

totalScore 1 and *totalScore 2* of the rule $r_5$as given in the section 5this parameter was fine tuned by algorithm 1 the below graph 1 represents the result of the above conducted experiment



Graph 1: stability analysis performance vs number of events

The next experiment is to know the impact of event history length with the suggested mechanism. This is based upon the work done [2].The dataset training component is stable without any major fluctuations between the rate and characteristics of intrusion. It gives a positive link between the performance and event history length graph 2 provides the information between the performance and event history length training set



Graph 2: performance vs. event history length

Where, *precision*: is the percentage of correctly incidental events relative to the total number of events incidental considering the Intrusion Detection System evaluation it may be referred as percentage of correctly inferred intrusion events relatively to all the events identified as intrusion.

Precision can be computed by the following formula $100 *\{ \frac{|\{number\ \ of\ relevant\ \ events\}\cap\{number\ \ of\ retrieved\ \ events\}|}{|\{total\ \ number\ \ ofretrievend\ \ \ events\}|}\}$

recall: is the percentage of correctly inferred events relative to the actual total number of events occurred in this time interval and regarding the intrusion detection system it may be defined as the percentage of correctly inferred normal events relatively to all the actual intrusion events.

recall can be computed from the below formula,

$100*\{ \frac{|\{number\ \ of\ releavent\ \ events\}\cap\{number\ \ of\ retrieved\ \ events\}|}{|\{umber\ \ of\ relevent\ \ events\}|}\}$

falseAlarm: to identify the percentage of incorrectly inferred events relative to the number of explicit events in the time interval

Performance evaluation is calculated by comparing the estimated event history $\hat{h}_{s(k)}^{f(k)}$ received from the inference mechanism and the actual event history $h_{s(k)}^{f(k)}$ provided by expert feed back at the end of time interval [formula] .A True Positive(TP) instance occurs whenever intrusion event was correctly inferred by the mechanism .A False Positive(FP) is identified whenever an event history is incorrectly inferred as intrusion ,when the lack of event was correctly identified as such ,it is referred to as True Negative(TN) and finally False Positive(FP) instance occurs whenever an event occurs yet was not inferred by the system.

10. CONCLUSION

The frame work devised is one of the solutions to the domain expert's complexity in upgrading the rules implemented for event processing and the efficiency of the event derivation under uncertainty the other possible ways of implementing this same frame work can be done in regression model or a model using machine learning techniques from large set of historical datasets. This approach is general suitable for enabling unpredictable conclusion for any event driven system. The complexity of the definition of rule parameter is been reduced by the Rule Prediction and Rule Correction paradigm the continuous tuning of the rule is carried by using the machine learning technique. This method can be achieved by other ways such as implementing a technique that will predict the efficiency of the initial value defined by the domain expert

11. FUTURE WORK

There is much other way in which one could attempt to directly model the rules using an extension of Bayesian networks or a stochastic extension of Petri Nets. Another possibility is the direct creation of statistical models (e.g., regression) or a model created using machine learning techniques from a large historical set of data This frame work can be implemented in a highly throughput system so that its performance can be increased

REFERENCES

[1] Efficient processing of uncertain events in rule based system SegevWasserkrug, Avigdor Gal, Senior Member, IEEE, OpherEtzion, and YuliaTurchin January 2012

[2] Y. Turchin, A. Gal, and S. Wasserkrug, "Tuning Complex Event Processing Rules Using the Prediction-Correction Paradigm," Proc. Third ACM Int'l Conf. Distributed Event-Based Systems (DEBS '09), 2009.

[3] R. Ammon, C. Emmersberger, T. Greiner, A. Paschke, F. Springer, and C. Wolff, "Event-Driven Business Process Management,"Proc. Second Int'l Conf. Distributed Event-Based Systems (DEBS '08),July 2008.

[4] M. Balazinska, N. Khoussainova, and D. Suciu, “PEEX: Extracting Probabilistic Events from Rfid Data,” Proc. Int’l Conf. Data Eng.(ICDE), 2008.

[5] K. Kersting and L. De Readt, “Bayesian Logic Programming,” An Introduction to Statistical Relational Learning, pp. 291-322, MITPress, 2007.

[6] S. Wasserkrug, A. Gal, O. Etzion, and Y. Turchin, “Complex EventProcessing over Uncertain Data,” Proc. Second Int’l Conf. DistributedEvent-Based Systems (DEBS ’08), pp. 253-264, 2008.

[7] C. Re, N. Dalvi, and D. Suciu, “Query Evaluation on ProbabilisticDatabases,” IEEE Data Eng. Bull., vol. 29, no. 1, pp. 25-31, Mar.2006.

[8] Active Database Systems: Triggers and Rules for Advanced Database Processing, J. Widom, and S. Ceri, eds. Morgan-Kaufmann, 1996.

[9] A. Gal and E. Hadar, “Generic Architecture of Complex EventProcessing Systems,” Handbook of Research on Advanced Distributed
Event-Based Systems, Publish/Subscribe and Message Filtering Technologies,A. Hinze and A. Buchmann, eds., IGI Global, 2009.

[10] P.R. Pietzuch, B. Shand, and J. Bacon, “Composite Event Detectionas a Generic Middleware Extension,” IEEE Network, vol. 18, no. 1,pp. 44-55, Jan./Feb. 2004.

[11] R. Puttini, Z. Marrakchi, and A Bayesian classification model for real-time intrusion detection.In 22th International Workshop on Bayesian Inferenceand Maximum Entropy Methods in Science and engineering 2002.

[12] R. Kalman. A new approach to linear _ltering and prediction problems. Journal of Basic Engineering, 82(1):35 - 45, 1960.

[13] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani A Detailed Analysis of the KDD CUP 99 Data Set, IEEE symposium on computational intelligence in security and defense application (CISDA 2009)