A Formal Framework for Intrusion Detection within an Information System based on Workflow Audit

Atsa Etoundi Roger
University of Yaounde I, Cameroon
Faculty of science
Department of Computer Science

Mboupda Moyo Achille
University of Yaounde I, Cameroon
Faculty of science
Department of Computer Science

Nkoulou Onanena Georges
University of Paris-Est Creteil
Faculty of science and technology

Nkondock Mi Bahanag Nicolas
University of Yaounde I, Cameroon
Faculty of science
Department of Computer Science

ABSTRACT

Nowadays information systems are very critical and important as their various users are distributed around the world. The role played by the information system in the evolution of the society has led to a new form of economy, the leaders in this economy are those who will be able to master their information system in terms of security issues. Based on the quality of data managed in the information system for decision making, the security issue becomes more and more crucial. Among the challenges that are faced in the information system, it appears that the intrusion detection problem is the major challenge that needs to be discussed first as all attacks start with the intrusion which precedes various malicious activities. Many works had been done in this domain but the intrusion detection problem is still an open research topic in computer science. In this paper, the described problem is considered as an engineering one. The approach used in this research is based on the workflow theory which allows carrying out an efficient identification in different activities that are able to be performed. The defined approach is focused on a formal and sound description of resources that participate in the execution of identified activities. The result of this paper is the definition of a formal framework for intrusion detection based on workflow execution analysis.

Keywords:

Intrusion detection model, Workflow execution audit, Information system audit, Activity categorization

1. INTRODUCTION

As information systems play increasingly vital roles in modern society, they have become the target of enemies and criminals. Therefore, there is a need to find all possible best ways to protect these information systems. The security of an information system is compromised when an intrusion takes place. Based on the quality of data stored and managed in the information system for decision making within an organization, the security issues become more and more crucial. Attacks in information systems are dangerous as atomic bomb. The illustration example is the attack of the Estonian system in 2007 by Russian hackers. To do that, they used a system called Botnet to diffuse spams to many important services (bank, government,. . .) and then saturated the Estonian network; knowing that Estonia is a country where everything is managed with technology, so nothing was available . Another example is the action of a virus called ”Stuxnet” created to affect Iranian nuclear program. It succeeds its task in 2010 by affecting processors used to control the rotation speed of centrifuges present in the nuclear station. ”Stuxnet” was introduced in many computers in such a way that 3 months after its diffusion, a very large scope of computers was infected. Thus, an infected usb flash which was used on the protected nuclear system by an authorize person, had corrupted the system. The processors was constraint by a malicious code to hide real information to administrators about centrifuges speed execution. Finally, in a few moment, all the centrifuges where destroyed and nuclear program halted there for 2 years.

In this case, it appears that a bomb was not built to attack Iranian nuclear but a virus called ”Stuxnet” acted more efficiently than a bomb. There exists some examples like that which show and justify the importance of security management. Among challenges that are faced in the information system, it appears that the intrusion detection problem [1] is the major challenge that need to be tackled first as all attacks start with the intrusion which precedes various malicious activities in the information system. An intrusion can be defined as any set of actions that attempt to compromise the integrity, confidentiality or availability of a resource [2]. Many works have been done in this domain but the intrusion detection problem is still an open research topic in computer science based on the huge number of workshops and conferences that are organized in this field. In the state of the art of the intrusion detection, many techniques have been defined such as Target Monitoring which uses cryptography algorithm that computes a cryptochecksum for each target file and each modification and changes such as file modification or program logon which would cause changes in the cryptochecksum are reported by the IDS; Stealth Probes [1] which collects and correlates data to try to detect attacks made over long period of time, often referred to as low and slow attacks, but there are mainly two types of intrusion detection techniques namely the Anomaly based intrusion detection and signature-based intrusion detection.

1.1 The anomaly detection technique

Anomaly detection is based on the normal behavior of an actor within an information system. Therefore, any action that significantly deviates from the normal behavior is considered like intrusion. Anomaly based intrusion detection is useful for detecting attacks like (1) misuse of protocol and service ports, in this case features of the standard protocols can sometimes be violated or modified by an intruder in order to tunnel through a firewall. An installation of backdoor services on well-known standard ports is another common misuse of service ports; (2) Denial of Service (DoS) based on crafted payloads, in this type of attack when a malicious intruder creates an attack using a crafted IP packet, the resulting DoS can occur on the network bandwidth, CPU cycles, memory resources, or application process/programs. The impact of the DoS attack is an anomaly in service quality; (3) Distributed Denial of Service (DDoS) is a form of attack that floods the network with a large volume of traffic. This is due to the fact that sophisticated attack traffic may not be distinguishable from regular network traffic on a peer packet basis and the attack does not manifest a specific signature that can be captured by signature-based mechanisms; (4) Buffer overflow which is the most common vulnerability exploited by attackers. Buffer overflow with shell code execution is the most serious form of this exploit because a successful attack can result in arbitrary program execution on the victim information system. Many exploited fields, such as user passwords for File transfer protocol, are supposedly made of printable ASCII characters based on the standard Request For Comments (RFCs) by the Internet Engineering Task Force (IETF). Excessive non-printable ASCII characters are anomalies of strong suspicion. Furthermore, shell code embedded in these fields are sure signs of malicious intent; (5) Other natural network failures which is based on the failures in routers/switches that can result in changes in traffic pattern observed at certain points of the network. This type of attack can be observed by a sudden dropping in the volume of traffic due to broken connections, or in the form of traffic shift from one link to another due to traffic rerouting as a recovery action. All these changes are noteworthy and can be detected as traffic anomalies. Anomaly detection technique represents a broad spectrum of detection techniques, for example Protocol anomaly detection, Application payload anomaly, and Statistical anomaly based intrusion detection [3, 4]. The anomaly based intrusion detection techniques allow to detect unusual behavior and thus have the ability to detect symptoms of attacks without specific knowledge of details. They also help in producing information that can be used to define signatures for misuse detectors. However, these techniques usually produce a large number of false alarms due to the unpredictable behaviors of users and networks; moreover, they often require extensive training sets of system event records in order to characterize normal behavior patterns.

1.2 The signature-based detection technique

The signature-based detection technique also known as misuse detection technique allows in detecting and catching intrusions in terms of the characteristics of known attacks or system vulnerabilities. Therefore, any action that conforms to the pattern of a known attack or vulnerability is considered as intrusive. This technique refers to techniques that use patterns of known intrusions or weak spots of a system to match and identify intrusions. The sequence of attack actions or activities, the conditions that compromise an information system's security, as well as the damage left behind by intrusions can be represented by a number of general pattern matching models. In order to carry out the detection of intrusion, researchers in this field use rules to describe attack actions [5], state transition diagrams to model general states of the system and access control violations, and Colored Petri nets to represent intrusion signatures as sequences of events on the target system. In the signature based intrusion detection, two techniques are used: (a) "Expression matching" is based on expression matching, which searches an event stream for occurrences of specified patterns or signatures; (b) "State transition analysis" models attacks as a network of states and transitions (matching events) where every observed event is applied to the finite state machine instances which represents each an attack scenario, possibly causing transitions. Any machine that reaches its final state indicates an attack. This approach allows complex intrusion scenarios to be modeled in a simple way, and is capable of detecting slow or distributed attacks, but may have difficulty in expressing elaborated scenarios. Based on these techniques, the signature-based intrusion detection technique allows in detecting attacks without generating an overwhelming number of false alarms as it was the case for the anomaly detection technique, they can also help information system manager in quickly and reliably diagnosing the use of a specified attack tool or technique in order to take corrective measures and track security problems on their information systems. However, the signature-based intrusion detection can only detect known attacks and needs to constantly update the information regarding new attacks. Therefore, newly invented attacks will likely go undetected, leading to unacceptable false negative error rates.

Based on these various techniques, many tools have been developed in order to face the intrusion detection problem depending on the application domain. Despite the popularity of the above techniques and associated tools, there is still enormous work to be carried out in the intrusion detection problem field as there is no solution that handles this problem in an entire manner. As the various techniques defined do not overcome all the difficulties encountered in the intrusion detection problem. In this paper, the intrusion detection problem is considered as an engineering one. The approach used in this research essay is based on the workflow theory which allows carrying out an efficient identification of different activities that are able to be performed within an information system. The proposed approach is focused on a formal and sound description of resources (equipments, programs, and human actor) that participate in the execution of identified activities. The result of this paper is the definition of a formal framework for intrusion detection based on workflow execution audit.

The rest of the paper is organized as follows: in section 2 a survey of a workflow mining is presented. This survey is followed by the section 3 where the definition of keys elements of the workflow theory that are suitable to build the proposed framework in section 4 are given. The section 5 concludes the paper by highlighting some futures works.

2. WORKFLOW MINING

The starting point for workflow mining is based on workflow log containing information about the workflow process as it is actually being executed in the information system. The workflow life cycle consists of four phases [5]: (1) the workflow design which allows the construction of the workflow model based on the information at hand and the goals to be achieved with the identified activities and associated constraints, (2) the workflow configuration which deals with limitation and particularities of the resulting workflow management system in the achievement of certain objectives within an organization, (3) the workflow enactment which deals with the integration in the used workflow system new functionalities according to new requirement that have been identified based on the strategy of the organization. In this phase, the information system is aligned to the vision of the organization, and (4) the workflow diagnosis in this phase from which data obtained from workflow instances execution are analyzed. This analysis can result to workflow/workflow process reengineering or provide inputs for the design of a new workflow which completes the live cycle of the former workflow.

To carry out the workflow enactment and workflow diagnosis tasks, the workflow mining is used as a guideline. The purpose of workflow mining is therefore to reverse the process and collect data at runtime to support workflow design and analysis. The information collected during the execution phase of a workflow is used to derive a model explaining the events recorded in the information system. The modeling of a workflow instance is usually carried out by considering the perception but does not really express what will exactly be done in practice. This looks to be the same in the security domain as the security policy is defined about the perception that the information system security manager has but not what is really carried out in the real world. For this purpose, models are often normative in the since that they state what should be done rather than what is really done by the workflow process. The type of models seems to be subjective. For a model to be objective, the designer must take into consideration data that are produced by the real execution of workflow instances. The data obtained are therefore linked to activities, time constraints and resources responsible for their generation. The workflow mining can then be very important in the intrusion detection problem as it allows drawing up what is really carried out in the information system by building a model corresponding on the exact situation or the state of the information system in terms of workflow execution, this model can therefore be compared to perceived model also known as an " priori" model in order to highlight the gap between the two models. The "a priori" model is generally based on the perceived security policy of the information which is defined by the set of actions to be performed by identified users based of given resources. The " priori" model is computed based on the information obtained within the execution phase of all activities in the information system. These associated data are stored in the information system logs. These logs therefore keep track of every event that occurs in the information system for future analysis and decision making. Closely monitoring the events taking place at runtime also enables detecting discrepancies between the design constructed in the design phase of the security policy and the actual execution registered in the intrusion handling. Workflow technology is moving into the direction of more operational flexibility to deal with workflow exception handling. Most exceptions can be seen as an intrusion in the intrusion detection problem. As a result, information system users can deviate from the " priori" information security policy design, for this purpose, according to the data kept in information system logs, the related deviation can be monitored. In order to push out the above defined gap for intrusion detection purpose. There is no uncertainty that workflow mining techniques can be used to create a feedback loop to adapt the workflow model to changing circumstances and detect in the mean time imperfections during and after the execution phase. The use of workflow mining in this work is not to show how workflow models are built from workflow event logs but detect intrusion from event logs based on the predefined information system security policy. More materials concerning workflow mining can be found in [6, 7, 8].

3. KEYS ELEMENTS

In this section, salient concepts suitable for the modeling of a workflow mining for intrusion detection problem are defined. The presentation of these concepts is given using the denotational semantics [4]. Concepts are modeled from the environment used by an activity for its execution to human actors in charge of this execution.

3.1 The Environment Description Model

In the modeling of business processes proposed by a great number of researchers, such as Van der Aalst [6] and Jorg Becker and al. [9], one can notice that they have neglected the effect that an environment can produce in the achievement of a given business goal within an organization. For example, in developing countries, moving from one hospital to another, the manner of carrying out surgeries differ even if the surgeons graduated from the same institute. This is due to the fact that hospitals are not equipped in the same manner. ln the modeling of a business process, the environment within which it will be performed has to be taken into consideration.

Without any loss of generality, an environment is assumed to be a set of different metrics whose value may change [10, 11]. These metrics are primitive Boolean observers denoted by Observer. The associated value of each observer depends on the current state of the environment.

Formally, an environment E is defined as a triple $\langle \Theta, S, val \rangle$ where:
- Θ is a non empty set of observers;
- S is a non empty set of states;
- $val : \Theta \to (S \to Bool)$ is a function which describes the behavior of observers.

In the rest of this paper, $val(o)(s)$ is denoted by $s(o)$ where s denotes a state and o an observer, $s(o)$ is the value of the observer o in the state s. Given a state s, the set of observers whose value is true defines the characteristic of s and is $S_c = \{o \in \Theta, s(o) = true\}$.

Given two states $s1$ and $s2$ of the set of states S of the environment E, the set of observers whose associated values are not the same is defined from the characteristics of the two states. This set is called gap between $s1$ and $s2$ and is denoted by $s1 \bullet s2 = (s1_c - s2_c) \cup (s2_c - s1_c)$.

Given an environment E, the observers in Θ define the alphabet that permits to reason about events that occur on E. The language defined from this alphabet is denoted by the set of conditions or formulae C. A condition $c \in C$ is an assertion over observers and is defined as a first order formula. The basic elements of C are therefore all the observers of Θ. The elements of C are formed by the following:

$$\begin{cases} \text{if} & \text{o} \in \Theta, \text{ then } o \in C \\ \text{if} & \text{o} \in \Theta, \text{ then } \neg o \in C \\ \text{if} & \text{o}_1, o_2 \in C, \text{ then } o_1 \wedge o_2, o_1 \vee o_2, o_1 \Rightarrow o_2 \in C \end{cases}$$

A condition c can be decomposed into a set of observers $+c$ whose values are evaluated to true and a set of observers $-c$ that are evaluated to false. The two sets do not have any common element i.e. $+c \cap -c = \emptyset$.

Given a condition $c \in C$ and a state s, c is satisfied within the state s if the result of its evaluation is true, i.e. $s(c) = true$.

3.1.1 State of an Environment. A state is a snapshot of an environment within a time. From this snapshot, facts are observed. Some of these facts or features of a state are true or false at this particular time. These facts are represented as some equivalent of predicate calculus formulae. Somewhat loosely, these facts and relations will be refer as attributes of a state. In a rigorous manner, let F be a set of formulae, and s be a state, then s is a subset of F i.e. $s \in F$.

In general, let S be a set of states, according to the definition of a state, $(S, \subseteq)$ is a partial ordered set. This paper doesn't dealing with any kind of set of states, it considers the set of states S having a least state, $\perp_S$, known as initial state of a business process or workflow from which the execution can be started. This initial state is therefore contained in all states of S i.e for all $s \in S$, $\perp_S \subseteq S$. In the meantime, S is required to have a least upper bound $\bigcup_S$ known as a state where the goal of the business process is satisfied. As such, the set of states of a business process is mapped to a CPO concept.

3.2 Task Description Model

A task is an atomic activity that cannot be split into smaller activities. The performance or execution of a task transforms the state of the environment into another state. A task is therefore an action within a state of an environment. Before a task can be executed, the state of the environment should satisfy a specific condition called pre-condition,and when this execution is completed another condition, called post-condition is satisfied. A task is formally defined by a triple $< nt, pre, post >$ where nt denotes the name of the task, pre its pre-condition, and $post$ its post-condition.

The action of a task within an environment is to transform its current state into a new one. When $< t, pre, post >$ is a task, s a given state where the precondition pre is satisfied i.e $s(pre) = true$, the action of t in the state s is the new state $t(s)$ which satisfies the post condition $post$ i.e $t(s)(post) = true$. In general, the action of a task t within the state s is characterized by the observers of s whose value has been modified.

DEFINITION 1. *(Task action)*

Let $E = \langle \Theta, S, val \rangle$ be an environment, s a given state and t a task whose pre condition is satisfied in s, then the action of t in s denoted by t_s and is specified by $t_s = \{o : \Theta, s(o) \neq t(s)(o)\}$.

When there will be no ambiguity, a task will be represented by its name t and $pre(t)$ respectively $post(t)$ will denote respectively its pre and post-condition. Based on the post-condition of a task t, and the state s where $s(post(t)) = true$, the conjecture $t_s = +post(t) \cup -post(t)$ can be done.

DEFINITION 2. *(Conflicting Tasks)*

The action of tasks within an environment can be conflicted as many tasks can modify the same observers at the same time. For this purpose, t_1 and t_2 are conflicting tasks in the state s, which is denoted by $overlap(t_1, t_2, s)$, if and only if:

$$\begin{cases} \text{s(pre(t}_1\text{))} = s(pre(t_2)) = true \\ +\text{post(t}_1\text{)} \cap -post(t_2) \neq 0 \\ +\text{post(t}_2\text{)} \cap -post(t_1) \neq 0 \end{cases}$$

DEFINITION 3. *(Shift)*

Let SoT be a none empty set of tasks and s a given state, a shift denoted by Shf is a couple $Shf = < s, SoT >$ composed with the state s and the set of non conflicting tasks SoT within s.

Formally, let $Shf = < s, SoT >$ be a shift, the following properties are satisfied:

$$\begin{cases} \text{SoT} \neq \emptyset \quad (1); \\ \forall t \in SoT,\ s(pre(t)) = true \quad (2); \\ \forall t \in SoT,\ t \neq t' \Rightarrow overlap(t, t', s) = false \quad (3) \end{cases}$$

Let $Sht = < s, SoT >$ be a shift, the simultaneous actions of SoT in s, denoted by $ts(s)$, is captured by the set of observers whose values are modified within s, that is: SoT(s)=$\bigcup \left\{ \begin{array}{l} \text{o} \in \Theta\text{:} \\ \text{o} \in -post(t_i) \end{array} \right\}$,t$_i \in SoT$

DEFINITION 4. *(Chain)*

A chain is an execution path of tasks according to their actions in states, their triggering conditions is denoted by $P = \prod_{i=1}^{n} Sht_i$, and is specified as a finite sequence of shifts where n represents the length of the sequence.

Let P be a path of length $n > 1$, and $sh_k = \langle\ s_k,\ st_k\ \rangle$, $shk + 1 = \langle\ s_{k+1},\ st_{k+1}\ \rangle$ notes respectively the shift in the range k and $k + 1$, the state s_{k+l} is the resulting state of the execution of the set of tasks st_k i.e $s_{k+1} = st_k(s_k)$. When there will be no ambiguity, the shift of the range k of the path P will be denoted by $P(k)$.

Let $Sht_k = \langle\ s_k,\ SoT_k\ \rangle$ and $Sht_{k+1} = \langle\ s_{k+1},\ SoT_{k+1}\ \rangle$ be two shifts where $Sht_k = SoT_k(s_k)$, the difference between the states s_k and s_{k+1} is denoted by $\overline{s_k + s_{k+1}}$ and is defined as follows: $\overline{s_k + s_{k+1}} = SoT_k(s_k)$

LEMMA 5.

Let p be an executin path and $t \in SoT(p(k))$ with $k \leq length(p)$ then there will always exist m such that $m > k$ and $S(p(m))(post(t)) = false$.

LEMMA 6.

Let p be an execution path then $SoT(p(length(p))) = \emptyset$

DEFINITION 7. *(State ordering)*

Let P be a path of length $n > 1$, $Sht_k = \langle\ s_k, SoT_k\ \rangle$ and $Sht_{k+1} = \langle\ s_{k+1}, SoT_{k+1}\ \rangle$ be two consecutive shifts in P with $k < n$ then $s_k \subseteq s_{k+1}$ specifies the fact that the set of observers modified in s_k after the actions of SoT are contained in the set of observers of s_{k+1} with the same values.

LEMMA 8.

Let P be an execution path, S the set of states of P, then $(S, \subseteq)$ is CPO where the least upper bound state in the last state of P and the least state is the first state of P.

At the level of this modeling, it is important to ensure that the execution of a task t will stop at a certain time. In order to do so, the set of observers that should be modified by t must be contained partially or totally in the observers forming its pre-condition $(-pre(t) \cup -pre(t)) \cap (-post(t) \cup +post(t)) \neq \emptyset$. From the definition of the execution path of tasks, the relation within the set of tasks T based on the set of states S can be specified. This relation is denoted by $\trianglelefteq$.

DEFINITION 9. *(Ordering of Tasks)*

Let T be a set of tasks, and t_1 and t_2 be two tasks of T, $t_1 \trianglelefteq t_2$ if and only if for all chain CH such that if n_{t_1} and n_{t_2} denote respectively the maximum range of t_1 and t_2 in CH, then $n_{t_1} \leq n_{t_2}$.

This relation has the following properties:

(1) *reflexivity*: $t \trianglelefteq t$ this simply means that the task t belongs to the chain CH;
(2) *antisymmetric*: if $t_1 \trianglelefteq t_2$ and $t_2 \trianglelefteq t_1$ in the chain D then $t_1 = t_2$. By convention, there will always exist a path from each task to itself;
(3) *transitivity*: obviously if in the chain CH, $t_1 \trianglelefteq t_2$ and $t_2 \trianglelefteq t_3$ then $t_1 \trianglelefteq t_3$

LEMMA 10.

The set of tasks T associated with the relation previously defined $\trianglelefteq$, i.e. $(T, \trianglelefteq)$, forms a complete partial ordered set.

3.2.1 Palette

DEFINITION 11.

Let E be an environment, and S be a set of different states that E may reach according to the actions of tasks T, then a palette P is a couple $< E, S \to S >$. In the rest of this paper, the set of functions $S \to S$ will be denoted by T, the set of tasks of the palette. When there will be no ambiguity, $P(E)$ and $P(T)$ will denote the environment and the set of tasks of the palette P respectively.

The actions of the set of tasks T of the palette P in the environment E are to change at least once the value of each observer of Θ in E. To this end, the consecutive actions of a non empty set of tasks within an environment may not modify all the observers in this environment. The set of observers whose value are not changed during the execution of any given none empty set of tasks will be abstracted from all the possible states of the environment, i.e. $\forall o \in \Theta \begin{cases} \exists\, t_1 \in T, \text{o} \in -post(t_1) \text{ or} \\ \exists\, t_2 \in T, \text{o} \in +post(t_2) \end{cases}$
Given a palette P, according to the environment changes within organizations and the different executions of tasks that can take place, different ways in which tasks can be executed have to be captured. In the rest of the paper, SPP will be use to specify the set of execution paths that can be obtained from a palette P.

LEMMA 12.

Let P be a palette, $s \in S(P)$ a given state of the environment $E(P)$ of P, there will always exist a path $p \in SP_p$ such that $s \in S(P)$, where $S(p)$ denotes the set of states of the path p.

LEMMA 13.

Let $P =< E, T >$ be a palette, and $t \in T$, there will exist an execution path $ch \in SP_p$ where SP_p denotes the set of possible execution paths of T, $ch(n) = \langle\, s_n, SoT_n \,\rangle$ such that $t \in SoT_n$.

3.3 Business Process Model

A business process is a collection of activities or tasks designed to produce a specific output for customers. It implies a strong emphasis on how work is done within an organization in order to deliver a particular service. A process is thus a specific order of work activities across time and space, with a beginning, an end, and clearly defined inputs and outputs. The output is the reason the organization does this work and is defined in terms of the benefits this process has for the organization as a whole.

DEFINITION 14. *(A service)*

A service is the characteristic of a business process and is defined as a composition of a set of criteria that characterize what is delivered within an organization, where each criterion is represented by an observer.

The model of a business process is defined as a couple $< P, G >$ where P is a palette and G the service to be achieved. According to the definition of the palette, the ordering of tasks is captured explicitly by their pre conditions and the states of the environment within which their execution is being carried out.

This approach reduces the number of patterns to be used in order to capture various ways tasks can be ordered. This is the main difference between the approach in this paper and other BPM theory papers presented in the literature. In these works, the Workflow Management Coalition [6] has identified four basic control structures for workflows: OR-SPLIT, OR-Join, AND-Split and AND-Join. More control structures have been identified by Van der Aalst in [7]. These control flow structures can be formally captured by the first order formula. A new direction for future works will consist of presenting these concepts using the denotation defined above.

LEMMA 15.

There will always exist a state S_{lub} such that when it is reached, other states cannot be reached. This state is called a least upper bound state of the associated business process.

LEMMA 16.

There will always exist a state S_{ini} from which the execution of the business process starts. This state is called a least state of the associated business process. For each service associated to a given business process, a set of qualities of service is defined to deal with the daily work and the competitive pressure of the network economy.

3.4 QoS Model

The quality of service denoted by QoS represents the performances of the service which determine the level of satisfaction projected for the recipients of the services. The level of satisfaction is defined as a set of properties, criteria, characteristics and performances of the services delivered to the customers. Several works are made in this field, each one defining a specific set of criteria specified in order to measure the QoS. In the literature, there is no consensus yet on the definition of the set of common criteria to evaluate the quality of service delivered in the organizations [12, 13]. The evaluation criteria are defined according to the objectives and specificities of each company. In this work, an abstract model which gives the semantics of the quality of service is defined.

DEFINITION 17.

Let Cr be a set of criteria considered in the evaluation of the quality of service, Val the set of values that can be assigned to these criteria, and f a map defined by $f : C \to Val$, the QoS is defined by (C, Val, f).

Given two QoS q_1 and q_2 such that $q_1 = (Cr_1, Val_1, f_1)$ and $q_2 = (Cr_2, Val_2, f_2)$, q_1 and q_2 are compatible and denote by $q_1 \triangle q_2$ if and only if $C_1 = C_2$ and $Val_1 = Val_2$. When q_1 and q_2 are compatible, q_1 is better to q_2 and denote by $q_1 \subseteq q_2$ if and only if $\forall c \in C_1, f_1(c) \leq f_2(c)$. In the rest of the paper, $(\Phi, \subseteq)$ will be use to denote the partial ordered set of compatible qualities of services.

DEFINITION 18. *(Well Defined Business Process)*

Let, $BP =< P, G >$ be a business process, BP is well defined if and only if all the observers that form its goal (service) are contained in the set of observers of the environment E i.e. $-G \cup +G \subseteq \Theta$ (E).

DEFINITION 19. *(Well Formed Business Process)*

Let $BP =< P, G >$ be a business process, BP is said to be well formed if and only if each execution chain SCH reaches the least upper bound state S_{lub} which satisfies the service G i.e.

$$\begin{cases} \forall ch \in SCH n_{ch} \in N, S_{lub} \in S \\ \text{n}_{ch} = length(ch) \\ \text{S}_{lub} \\ \text{S}_{lub}(G) = true \end{cases}$$

More formally, let SCH be the non empty set of different chains that can be obtained from a business process BP, and $CH \in SCH$ with the length n_{CH} such that the n_{CH}^{th} state S_{lub} of CH satisfies G i.e. $S_{lub} = true.$

DEFINITION 20. *(Deadlock- and Livelock-Free)*

Let BP be a business process, BP is deadlock- and livelock-free if and only if it guarantees that every execution chain reaches its least upper bound state satisfying the goal of the business process BP.

THEOREM 21.

Let BP denote a business process such that BP is well defined and well formed, then BP is deadlock- free and livelock-free.

The proof of this theorem can be found in [10].

All the execution paths of a business process start from the same state denoted by S_{ini}. It can be easily being shown that the set of states S_{BP} associated with the ordering relation $\subseteq$ as defined previously is CPO.

3.5 Agent Generic Model

There are many types of agents participating in the processing of tasks within an information system. An information system dealing with the processing of tasks is a hybrid system including hardware components with embedded software, human actors interacting with the hardware and software. Agents are performing tasks in order to carry out certain missions, which, in its turn, add a dimension to the quality of service. Each agent has a skill which is characterized by (Sk, Tks, mch) where Sk is the set of competencies, Tks the set of tasks and mch a map that gives for each competence $cp \in Sk$ the set of tasks $mch(cp) \in Tks$ that can be processed based on cp with $mch(cp) \neq \emptyset$. When there will be no ambiguity, the structure (Sk, Tks, mch) will be represented by Sk. Based on the organization put in place, the set of tasks performed by an agent is kept in a diary.

A diary is described by the set of tasks and the set of time intervals within which there are processed. Let $Pds = (TI, \subseteq, \cap, \triangle)$ be a set of time intervals such that $(TI, \subseteq)$ is a partial ordered set with ∂ the smallest time interval, $\cap$ and $\triangle$ be two maps defined as follows $\cap : TI$ x $TI \to TI$ and $\triangle : TI$ x $TI \to Boolean$. If p_1 and p_2 are two time intervals, p_1 and p_2 overlapped if and only if there exists a time interval p_3 such that: $\text{p}_1 \cap p_2 = t_3 \Rightarrow \begin{cases} \text{p}_3 \triangle p_1 \wedge p_3 \triangle p_2 \\ \text{p}_3 \subseteq p_1 \wedge p_3 \subseteq p_2 \end{cases}$ where $\cap$ and $\triangle$ define respectively the intersection and the overlapping relationship. When there will be no ambiguity, the set of time intervals will be represented by Pds. Based on the concepts of tasks and time interval, the diary concept is modeled by $< Tks, Pds, g >$ where Tks is the set of tasks, Pds the set of associated time intervals, and g a map defined by $g : Tks \to Pds$ such that $\forall\, t_1, t_2 \in Tks, t_1 \neq t_2 \Rightarrow \neg(g(t_1)\triangle \text{g}(\text{t}_2))$.

DEFINITION 22. *(Resource model)*

A resource model Rm is defined by $< Id, Sk, Ex, Dy, f >$ where Id is its identification, Sk, its set of skills, Ex, the set of associated experiences, Dy its associated diary and f a map which defines for each skill $sk \in Sk$ its associated experience $f(sk) \in Ex$.

3.6 Enterprise

An enterprise is a system dealing with the service delivery based on a certain quality of service. This system is organized in terms of business processes that are carried out, agents in charge of the processing of the associated tasks, and the resulting workflows. An information system IS is modeled by $(Io, BPs, Emps, WFs)$ where Io is its identification, BPs is the set of its business processes that can be run, $Emps$ its set of agents who participated in the processing of tasks, WFs its set of workflows.

A workflow is formally defined by $(Ts, Es, Ps, h, g, q, Q)^+$ where Ts is the set of none conflicting tasks, Es the set of agents dealing with the processing of Ts within the time intervals Ps to obtain the quality of service Q, h is the map $Ts \to Ps$ which defines for each task t, its time interval $h(t)$ within which it is processed, g a map $Es \to Ts$ which gives for each agent ag the associated performing task $g(ag)$, and q a map $Es \times Ts \to Q$ which gives for a given agent ag and its associated task tk the quality of service $g(ag, tk)$ obtained after the execution.

Based on the agents using or working in a given enterprise and their availability and the services required by customers, agents involved in different workflows associated to a business process will not necessarily be the same. To this end, according to their skills, the quality of service delivered may be different. The criteria for the evaluation of the quality of service will then sometimes be associated with minimum values when tasks will be processed by agent with minimum experience. More-over these values will be maximal when agent with maximum experience has been involved in the processing of tasks. The set of quality of service associated to a given business process will therefore have two specific qualities of service Q_{min} and Q_{max} which have the following properties.

LEMMA 23.

Let $Q_{min} = (C, Val, f_{qmin})$ and $Q_{max} = (C, Val, f_{qmax})$, be minimal and the maximal quality of service of a business process (P, Φ) then $\forall p = (C, V, f_p) \in \Phi$, $c \in C$, $f_{qmin}(c) \leq f_p(c)$, and $\forall q = (C, V, f_q \in \Phi$, $c \in C$, $f_q(c) \leq f_{qmax}(c)$.

4. FORMAL FRAMEWORK

In this section, the proposed framework for the intrusion detection problem within an information system is presented based on the workflow theory presented in the previous section. The construction of the model starts with a generic formal description of various resources that are founded in the information system suitable to be subject to attacks. This description is followed by the definition of the normative model of the security policy applied in the information system. The normative model is followed by the definition of the descriptive model associated to the effective usability of different resources of the information system. The section ends with the definition of an approach defining the intrusion detection based on the audit of workflow instances execution within the information system. The audit compares data related to the normative and descriptive models.

4.1 Information System Model

As defined in the introduction of this work, a resource within an information system is either a human actor, equipment, a function (program) or a data. Resources are using other resources in order to perform specific activities within some resources. Activities are different tasks or functions that can be carried out by resources within the information system. Each resource is associated with a set of activities that it may perform, and a set of activities to which it can be applied. Moreover, from a given resource, a set of traces of activities to which it can be applied, and the set of traces of activities that it performs are specified. In the same manner, a resource may contain sub-resources. Each sub-resource in a given resource is considered as a resource. The set of traces actually defines the so called event logs used in workflow mining.

DEFINITION 24. *(Resource)*

Formally, a resource, denoted by R in this work, is modeled by $< Id, Log\_in, Log\_out, Task\_in, Task\_out, SubRes >$ where:
- Id is the identifier of the resource;
- Log_in represents the event log of incoming activities;
- Log_out represents the event log of out coming activities;
- $Task\_in$ represents the set of possible in coming activities;
- $Task\_out$ represents the set of possible out coming activities;
- $SubRes$ denotes the set of sub resources.

When there will be no ambiguity for a given resource R, the following expressions will be equivalent:
- $R(Id) = Id$;
- $R(Log\_in) = Log\_in$;
- $R(Log\_out) = Log\_out$;
- $R(Task\_in) = Task\_in$;
- $R(Task\_out) = Task\_out$;
- $R(SubRes) = SubRes$.

Given a resource in an information system, its sub-resources are also resources of that information system. In this modeling approach, two type of resources are considered intermediary resources and terminal resources. A resource r is said to be intermediate when r has one or many sub-resources. Otherwise, r is called a terminal resource.

Resources are any agent that execute or able to participate in the achievement of an activity in the information system. Based on the fact that an activity itself is able to launch the execution of another activity, it is also considered to be the case among resources of the information system. Thus for a task t to be performed, many agents are considered. For this end, the different tasks that a resource can perform or trigger are based on the environment of the information system.

DEFINITION 25. *(Environment)*

The environment is defined by $E_{IS} =< \theta_{IS}, S_{IS}, Bool >$ Where:
- θ_{IS} is the set of observable events ($\theta_{IS} = \bigcup_{i=0}^{\infty}(Ev_i)$);
- S_{IS} is the set of states.

DEFINITION 26. *(Event)*

An Event, denoted by Ev, is modeled by $< Id, Task, T\_In, T\_Out, Resp\_Res, Used\_Res, Task\_Act >$ where:
- Id is Event identifier;
- $Task$ represents activity of the event;
- T_in is the beginning time of the event;
- T_out is the ending time of the event;
- $Resp\_Res$ is the responsible of resource used;
- $Task\_act$ represents the action of the task.

The execution of a given activity (task) is based on a specific state of the information system. A state S is formally modeled by the following $S = \bigcup_{r \in R}(S_r)$ where $S_r = Log\_in(r) \cup Log\_out(r)$.

Having defined the environment, the state and the resource model, one can easily deal with the information system.

DEFINITION 27. *(Information system)*

The information system based on all previously defined elements is represented by:
$IS = (R, S, BPs, WFs, BPs^{WFs}, Id\_Res^{Log\_in}, Id\_Res^{Log\_out}, Id\_Res^{Tr}, Id\_Res^{Task\_in-set}, Id\_Res^{Task\_out-set})$
where:
- R: the set of resources;
- S: the set of states;
- BPs: the set of business processes that can be supported in SI;
- WFs: the set of workflows that can be run in SI;
- $BPs^{WFs} = BPs \underset{m}{\longrightarrow} WFs$: the set of function that give for every business process the set of associated workflows;
- $Id\_Res^{Log\_in} = Id\_Res \underset{m}{\longrightarrow} Log\_in$: the set of functions define for a given resource its associated supported events;
- $Id\_Res^{Log\_out} = Id\_Res \underset{m}{\longrightarrow} Log\_out$: the set of functions that define for a given resource its associated initiated events;
$Id\_Res^{Tr} = Id\_Res \underset{m}{\longrightarrow} Tr$: the set of functions that define for a given resource its triggering conditions;
- $Id\_Res^{Task\_in-set} = Id\_Res \underset{m}{\longrightarrow} Task\_in - set$: the set of function that define for a given resource the set of tasks that it can handle;
$Id\_Res^{Task\_out-set} = Id\_Res \underset{m}{\longrightarrow} Task\_out - set$: the set of function that define for a given resource the set of tasks that it can perform.

The defined model of the information system does not take into consideration the way different activities will be carried out based on the usability of various resources. This lets the resulting model exposed to malicious operations. In order to avoid this, the security policy on how tasks and resources are put together in the achievement of various goals is needed. The requirement is discussed in the following section.

4.2 Security policy

The security policy within an information system is so important that it guides how each resource should be used in the resulting organization. This policy is a set of rules that are required to be followed by any known and authorized entity in the organization. In this work, based in the intrusion detection problem which is considered as an engineering one, the security policy is based on four main concepts i.e the *Resources*, the *Tasks*, the *Time* and the *Trigger*. The concept of resource is used in order to list all the resources of the organization whose usability can violate its security rules. The task concept allows to identify various activities, functions or tasks that can be executed anyhow in the organization based on identified resources. When a given activity is then executed with the organization, the trace of this execution should be kept for further investigation. For this purpose, one may like to know the time within which an observed execution has taken place. The answer of this important question is based on the definition of the starting and ending time of this execution. The *Period* concept related to the time concept is therefore very critical in this situation. In the management of the information system, some resources are keeping others ready to be used i.e if the resource kept is for example a program installed within a machine, it can be run at any time by another resource. this means that that this program has the required environment for its execution available. In order to control the execution of such resource or activity, one needs to a triggering condition that will really trigger the resulting execution. This also means that, an activity can have its pre condition satisfied, but if the triggering condition is not satisfied, this activity will not be able to be executed.Based on these defined concepts, the security policy SP is modeled as follows:

$$\text{SP} = \begin{cases} \text{out} = \text{R} \underset{m}{\longrightarrow} Task - set \\ \text{in} = \text{R} \underset{m}{\longrightarrow} Task - set \\ \text{p\_out} = \text{R} \underset{m}{\longrightarrow} R \underset{m}{\longrightarrow} Period \underset{m}{\longrightarrow} Task - set \\ \text{p\_in} = \text{R} \underset{m}{\longrightarrow} R \underset{m}{\longrightarrow} Period \underset{m}{\longrightarrow} Task - set \\ \text{Tg} = \text{Task} \underset{m}{\longrightarrow} Trigger \end{cases}$$

The various maps defined the relationships between the four concepts in order to highlight different events that took place in the information system by violating its security policy. Based on the security policy, the normative model of the information system can now be expressed.

4.3 Normative Information system model

Based on the application of the security policy in the target information system, and without any external action that that violate this policy the model of the information system within the time is as expected by the different managers in the organization. The Normative Model, denoted by IS_{nor}, is formally specified as follows
$IS_{nor} = (IS; Runs, SP)$
where:
- IS is a target information system;
- $Runs$ is the set of workflows that have been executed in IS by applying the security policy previously defined;
- SP is the associated security policy.

The normative model is far to match the reality of the information system. If one considers the higher number of attacks that faced different information systems around the world, the normative model of the information system is a view of spirit as each time, security rules are violated based on internal or external actions. The number of attacks reported is not the real situation since in many organizations, managers do not reveal attacks to the public. This is sometimes due to the fact that they do not want to frustrate theirs customers. As a consequence, it lead to the anger of customers, their weight bill which depends on the faithful relationship with their customers will decrease.In most of the time, they did not even know that their information system has been a source of attacks. Based on this strong remarks of the normative model, the real situation of the information system is represented by the descriptive model

4.4 Descriptive security model

This Model describes the information system using real activities execution within the system. Let IS_{des} be the Descriptive Model defined as follow:
$IS_{des} = \langle IS, Runs \rangle$
where:
- IS is the information system;
- $Runs$ is the set of events satisfied by triggering conditions for task execution.

4.5 Intrusion detection model

Let IS_{des} and IS_{nor} denote respectively the descriptive and the normative model of the information system IS, IS_{des} is said to be conform to the normative information system, noted $IS_{nor} \models IS_{des}$ if and only if for each resource r the following constraint is satisfied:

$log\_y(r, IS_{des}) \subseteq log\_y(r, IS_{nor}) \wedge log\_y(IS_{des}) \subseteq log\_y(IS_{nor})$

where $log\_y(r, IS_x)$ denotes the events log associated to the resource r in the (Normative or Descriptive) information system, with $y = in \mid out$ and $x = des \mid nor$. $log\_y(IS_{des}$ represents the set of events logs of the(Normative or Descriptive) information system.

In the same manner, the descriptive model IS_{des} will not be conformed to the normative model IS_{nor}, this is denoted by $IS_{nor} IS_{des}$, if and only if there exist a resource r in the Descriptive information system IS_{des} with an activity that has been executed by violating the security policy rule(s) in a periode of time $Period$ defined by $[T\_in, T\_out]$.

LEMMA 28.

Let $p1$ and $p2$ be to two Period; $p1 \models p2$ if $p1[T\_in] < p2[T\_in]$ and $p1[T\_out] > p2[T\_out]$.

An activity tk has been executed with the violation of the security policy if and only if one of the following constraints is satisfied:

$$\exists k \in E\_IS_{des} / \begin{cases} \text{Task (log\_x(r, IS}_{des})[k]) = tk \wedge tk \notin task\_x(r), \\ x = out \mid in \\ \text{P\_x(task (log\_x(r,IS}_{des})[k])) = pk \wedge pk P\_x(r), \\ x = out \mid in \\ \text{trigger((Task, IS}_{des})[k]) = tgk \wedge tgk \notin Tg(Task) \end{cases}$$

where:
- $Task(log\_x(r, IS_{des})[k])$ is the activity associated of the resource r that was recorded in the log_x through the Descriptive information system IS_{des} reported by the event k;
- $task\_x(r)$ is the set of activities that the resource r can initiated or support;
- $P\_x(task(log\_x(r, IS_{des})[k]))$ is the period of time in which activity runs;
- $P\_x(r)$ is the period of time that a resource r can run;
- $trigger((Task, IS_{des})[k])$ denote trigger for the Task associated to the event k in the Descriptive information system;
- $Tg(Task)$ is the set of triggers associated with all Task in IS_{des}.

DEFINITION 29.

Let IS denote an information system, r a resource and log_in and log_out it log files. The intrusion detection denoted by ID is defined as follow:
$ID = \bigcup_{i=1}^{m} (log\_in(r_i, IS_{des}) \setminus \text{log\_in}(\text{r}_i, IS_{nor}) \cup log\_out(r_i, IS_{des}) \setminus \text{log\_out}(\text{r}_i, IS_{nor}))$

THEOREM 30.

Let IS be an information system, if an event e of IS does not match it secure policy then, $e \in ID(IS)$.

DEFINITION 31.

Let IS_{nor} denote a normative information system, $log_{x=in|out}(r, IS_{nor}) = \{x \mid x$ is an event of IS and x match the security policy $\}$.

LEMMA 32.

Let e be a event and SP the security policy of the IS. e is said to match SP if and only if $e = \langle Task, T\_in, T\_out, Resp\_res, Used\_res, Task\_out\rangle$ with the following condition:

$$\begin{cases} \text{Task can run if and only if it triggering condition is satisfied} \\ \text{Task can run if and only during a time period T\_in and T\_out} \\ \text{Resp\_res and Used\_res can be used by Task only during a time} \\ \text{period T\_in and T\_out} \\ \text{The action of a task on Resp\_Ress or Used\_ress belong to Task\_act} \end{cases}$$

DEFINITION 33. *(Information system transition function)*

It is a function that allows events to transform a normative model to the associated descriptive model according to the security policy. This function is defined as follows:
$f : Event$ X $IS \rightarrow IS$ / If e match SP then $e \notin ID(IS)$ else $\in ID(IS)$

5. RELATED WORKS

Many modeling approaches have been developed in order to tackle the above defined problem. Some of the resulted models are based on genetic algorithms [14, 15, 16], while others are based on data mining [17, 18, 19]. Despite the popularity of these techniques, the proposed solutions are far from meeting the target as long as the information security is concerned. The above mentioned solutions do not take into consideration different types and abstractions of various resources that are found or act in the information system. Therefore, the associated solutions cannot efficiently handle the information system security policy in order to identify or prevent intrusion. It is possible to efficiently de tect or prevent intrusion when the description of every resource is defined. This allows defining the normative security model of the information system that should next be compared to the descriptive one in order to define the gap. This gap is called intrusion that needs to be detected or prevented. Based on the security policy representation and different resource models, the proposed approach in this paper deals with the definition of the so called gap.

6. CONCLUSION AND PERSPECTIVES

This paper provides a formal framework modeling intrusion detection within an information system. The workflow mining theory to be aware of the system evolution by analyzing its resources workflows events logs to detect intrusions. After presented two main approaches used for intrusion detection, one can defined some concepts manipulated in information systems to be able to set up the model used to detect intrusion especially thanks to events logs. Thus, using the proposed framework to detect attacks and then stoping them is not easy and it involves a deep knowledge in the domains of workflow mining and information systems security. Two challenge for future works are highlighted here. The first consists on the construction of an efficient tool based on the proposed model to handle intrusions within a given information system. The second deals with the definition of a digital forensic framework that could help to investigate after an attack in an information system detected by an intrusion detection engine.

7. REFERENCES

[1] Karthikeyan .K.R and A. Indra: "Intrusion Detection Tools and Techniques - A Survey". International Journal of Computer Theory and Engineering, Vol.2, No.6, December, 2010.

[2] Heady, R., G. Luger, A. Maccabe, and M. Servi lia: "The Architecture of a Network Level Intrusion Detection System". Technical report, Computer Science Department, University of New Mexico, 1990.

[3] Lunt, T.: "Detecting Intruders in Computer Systems". In: Proceedings of the 1993 Conference on Auditing and Computer Technology, 1993.

[4] Kumar, S. and E. H. Spafford: "A Software Architecture to Support Misuse Intrusion Detection". In: Proceedings of the lSth National Information Security Conference, 1995.

[5] A. Kartit, A. Saidi, F. Bezzazi, M. El marraki, A. Radi: "A new approach to intrusion detection system". Journal of Theoretical and Applied Information Technology, 29th February 2012. Vol. 36 No.2, ISSN: 1992-8645, www.jatit.org, E-ISSN: 1817-3195.

[6] W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster: "Workflow Mining: Discovering Process Models from Event Logs". IEEE Transactions on Knowledge and Data Engineering (TKDE), Accepted for publication, 2003.

[7] W.M.P. van der Aalst, B.F. van Dongen, J. Herbst, L. Marusterl, G.Schimm, and AJ.M.M. Weijters: "Workow Mining: A Survey of issues and Approaches". Data and Knowledge Engineering, Accepted for publication, 2003.

[8] Ilgun, K., R. A. Kemmerer, and P. A. Porras: "State Transition Analysis: A Rule-Based lntrusion Detection Approach". IEEE Transactions on Software Engineering 21(3), 1995.

[9] Jorg Becker, Patrick Delfmann: "Reference modeling: Efficient Information System Design Through Reuse of Information Models". Kindle Edition, Jul 2007.

[10] Atsa Etoundi Roger: "ATSERO Method: A Guideline for Business Process and Workflow Modeling Within an Enterprise ". International Journal of Scientific Engineering Research, Dec 2011.

[11] Bob Glushko: "Process Modeling for Information System Design". Oct 2008.

[12] Sandy Kemsley: "Business Process Modeling". TIBCO Software Inc, global headquarters, 3303 hillview avenue, Palo alto, ca 94304, 2011.

[13] Rafael Accorsi, Thomas Stocker, Gnter Mller: "On the Exploitation of Process Mining for Security Audits: The Process Discovery Case". SAC '13 Proceedings of the 28th Annual ACM Symposium on Applied Computing, Pages 1462-1468, ACM New York, NY, USA 2013.

[14] Mohammad Sazzadul Hoque, Abdul Mukit, Abu Naser Bikas: "An implementation of intrusion detection system using genetic algorithm". International Journal of Network Security and Its Applications (IJNSA), Vol.4, No.2, March 2012.

[15] Bharat S. Dhak, Shrikant Lade: "An Evolutionary Approach to Intrusion Detection System using Genetic Algorithm". International Journal of Emerging Technology and Advanced

Engineering. ISSN 2250-2459, ISO 9001:2008, Certified Journal, Volume 2, Issue 12, December 2012.

[16] A.A. Ojugo, A.O. Eboka, O.E. Okonta, R.E Yoro, F.O. Aghware: ”Genetic Algorithm Rule-Based Intrusion Detection System (GAIDS)”. Journal of Emerging Trends in Computing and Information Sciences, VOL. 3, NO. 8 Aug, 2012 ISSN 2079-8407.

[17] Qinglei Zhou, Yilin Zhao: ”The Design and Implementation of Intrusion Detection System based on Data Mining Technology”.Research Journal of Applied Sciences, Engineering and Technology 5(14): 3824-3829, 2013 ISSN: 2040-7459; e-ISSN: 2040-7467, Maxwell Scientific Organization, 2013.

[18] Yogita B. Bhavsar, Kalyani C.Waghmare: ”Intrusion Detection System Using Data Mining Technique: Support Vector Machine ” International Journal of Emerging Technology and Advanced Engineering, Website: www.ijetae.com, ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 3, Issue 3, March 2013.

[19] Parekh s.p, Madan b.s, Tugnayat r.m: ”Approach for intrusion detection system using data mining”, Journal of Data Mining and Knowledge Discovery ISSN: 22296662, and ISSN: 22296670, Volume 3, Issue 2, 2012, pp.-83-87.