International Journal of Communication Networks and Information Security (IJCNIS) Vol. 4, No. 2, August 2012

Process Aware Host-based Intrusion Detection Model

Hanieh Jalali[1], Ahmad Baraani[1]

[1]University of Isfahan, Computer Department,
Isfahan, Iran
{jalali, ahmadb}@eng.ui.ac.ir

***Abstract*:** Nowadays, many organizations use Process Aware Information Systems (PAISs) to automate their business process. As any other information systems, security plays a major role in PAIS to provide a secure state and maintain the system in it. In order to provide security in a PAIS, a Process Aware Host-based Intrusion Detection (PAHID) model is proposed in this paper. The model detects host-based intrusions in a PAIS using process mining techniques.

The proposed model uses both anomaly detection and misuse detection techniques for more efficiency, and organizational perspective of process mining is considered (rather than control-flow perspective) to detect more attack types. The model is automated and can deal with large logs and is suitable for flexible application domains. The PAHID model is implemented by the use of ProM framework and Java programming. It is evaluated by using a simulated log based on a real-world organization information system. Results demonstrate that the model provides high accuracy and low false positive rate.

***Keywords*:** Host-based Intrusion Detection, Process Aware Information System, Process Mining, Anomaly Detection, Misuse Detection**.**

1. Introduction

Recent Management trends largely motivated organizations to use Process Aware Information systems (PAISs), because of the fact that the processes move and change the data in an information system. Using PAISs leads to a shift from data-centric to process-centric systems. Therefore, business process logic becomes completely separated from application programs, also redesigning and extending of process models becomes easy. In some application domains, the information system must respond rapidly to new process models, or a defined business process model is not completely known before execution. Consequently, in these domains, normative PAISs like Workflow Management Systems are not suitable and a flexible PAIS is needed.

Considering any types of information systems, security plays an important role in. It is very difficult to provide a secure information system and maintain it in such a secure state during its lifetime and utilization. Therefore, there is a need for mechanisms that provide security in the information systems; one of the most important mechanisms is Intrusion Detection System (IDS). Intrusion Detection Systems monitor the usage of information systems or networks to detect any insecure state. They detect misuse attempt or action executed by authorized users or unauthorized users that try to abuse their privileges or exploit security vulnerabilities [1]. Two types of IDSs have been declared: Network-based Intrusion Detection System (NIDS) and Host-based Intrusion Detection System (HIDS). Today, because of the limitations of NIDSs and use of lots of encryption methods in communication, detection domain has moved from networks to host systems. In order to provide security in the PAISs, we propose a host-based intrusion detection model.

On the other hand, there is an absolute need in the PAISs for auditing systems and techniques to extract knowledge from information recorded by the system. Moreover, the vast growth of log data in the form of audit trail, transaction logs, and data warehouses have resulted in the development of process mining techniques. The goal of process mining is to extract knowledge from event logs produced by information systems. Therefore, we propose a model for host-based intrusion detection in Process Aware Information Systems using process mining techniques.

Recent challenges, in the field of intrusion detection in PAISs, are few in number and limited to anomaly detection in the control-flow perspective of process mining. [2] presents two methods to detect anomalous traces in a PAIS. In this work, there is a need to have a known normal log, so this approach is not suitable for application domains that need flexible support, because a normal log is not known before execution. Other techniques to detect anomalous traces are presented in [3]-[6]. These techniques are suitable for flexible application domains. However, [3] and [4] cannot deal with large logs, because of adopted process mining and [5] needs a precise appropriateness metric to select an appropriate model and also, an automated solution might be implemented. [6] is a well-defined automated genetic-based solution for anomaly detection in PAISs, but it is limited to control-flow perspective.

Considering previous works, the design goal is to propose an automated solution for host-based intrusion detection in PAISs that provides a desired level of tradeoff between flexibility and security, and can deal with large amount of logs. In addition, despite previous research works that only consider control-flow perspective, organizational perspective are also considered for intrusion detection.

With these design objectives in mind, a Process Aware Host-based Intrusion Detection model (PAHID model) is proposed. In this model, it is tried to use both anomaly detection and misuse detection techniques to provide more efficiency. In addition, Intrusions in organizational perspective are also considered to detect more attacks. The proposed model is implemented using ProM framework and Java programming. ProM is an open-source plug-able framework that provides a wide range of process mining techniques. The PAHID model is evaluated through a simulated log of a real-world organization to demonstrate its efficiency and accuracy. Results show that the PAHID model has low false positive rate and high accuracy and its performance is better than previous works.

The rest of the paper is organized as follows. In section 2, the related works are reported with details. The characteristics of PAHID are described in detail in section 3. Section 4 presents the experimental result of the proposed model. Conclusion and future work are given in section 5.

2. Related Works

In order to detect intrusions in PAISs, some research works have been done that develop anomaly detection methods [2-6]. In [2], the authors present two methods based on α algorithm for detecting anomalous traces. These methods construct a normal process model based on a previously known normal log. Then, the conformances of new event traces, recorded in a separate log, are checked with the normal process model to discover anomalies. However, these methods are not suitable for application domains that need flexibility, because a normal log is not known before execution in these domains.

In other research works, the normal model is discovered during an anomaly detection process, so they are suitable for flexible application domains. In [3] and [4], three methods are presented and compared: sampling, threshold, and iterative. Nevertheless, these methods have some practical limitations to deal with larger logs, because of adopted incremental process mining algorithm. In [5], the authors present an approach for anomaly detection based on a formal definition of anomalous trace. The anomalous trace is defined through two parameters: (I) fitness model degree (p%); and (II) appropriateness of model (a). Using two parameters, an appropriate process model is discovered and every trace not fitting that model is considered as an anomaly. However, as pointed out by the authors, the approach needs a precise appropriateness metric to select the appropriate model; also an automated solution is needed. In [6], a model is presented for detecting anomalous traces based on genetic process mining. Because of using genetic process mining, the model is automated and does not have any practical limitations. This model is a suitable solution for anomaly detection in PAISs. Nevertheless, [6] like other research works is limited to control-flow perspective. Therefore, an attack may follow a normal path but it is executed by unauthorized users or roles, while it is not detected. Moreover, anomaly detection techniques may have high false positive rate.

The PAHID model uses both anomaly detection and misuse detection techniques to provide lower false positive rate and higher accuracy. Intrusions in organizational perspective are also detected in misuse detection phase. Furthermore, because of using genetic process mining, the model is automated and flexible and can deal with large logs.

3. Process Aware Host based Intrusion Detection Model

Intrusion detection is the process of monitoring events occurred in a computer system or network and analyzing them for signs of intrusions. An intrusion is defined as an attempt to compromise Confidentiality, Integrity, and Availability, or to bypass security mechanisms of a computer or network. Intrusions are caused by intruders accessing the system through the internet, authorized users that try to gain unauthorized additional privileges, or authorized users that try to misuse their given privileges. Intrusion Detection systems are hardware or software products that automate the process of monitoring and analyzing [7].

There are two main types of IDSs acting on different set of data: Host-based IDS and Network-based IDS. HIDS was the first intrusion detection domain. It is an application program installed on a host monitoring and analyzing network packets, logs, and events executed by application programs and operating systems to discover intrusions. NIDS is a commercial product installed on a special hardware and placed on a network node. It captures and analyses local network packets that go through the node for discovering intrusions. In the last decade, NIDSs have clearly dominated HIDSs. However, by increasing the use of the fast Ethernet cards and encryption of network packets data, NIDSs are encountered with some problems. They cannot provide high accuracy and low false positive rate, so the detection domain has once again moved back to host systems, where the content data is clearly visible and the quality of logs provided by operating systems and application programs are high. Host-based IDSs analyze the information provided by a single computer system, so they can analyze events with high accuracy and reliability, and discover exactly which processes and users are involved in an attack. With these advantages, we propose a model for host-based intrusion detection in PAISs.

In IDSs, there are two primary techniques for analyzing events and detecting attacks: anomaly detection and misuse detection. Anomaly detectors identify abnormal unusual behaviors (anomalies) on a host or network. They work with this assumption that attacks are different from normal (legitimate) behaviors, so they can be detected by systems that identify these differences. Anomaly detectors construct profiles or models that represent normal behavior of users, hosts, or network connections using data collected during a previous normal operation of the system. Then, they collect current event data and use different measures to detect any deviation from normal behavior. IDSs based on anomaly detection can detect unknown attacks without any special knowledge of details. However, they produce high false positive rate due to the unpredictable behavior of the users and systems, also they need extensive training sets of normal behavior of system to construct the normal profile or model. In misuse detection technique, activities are analyzed in order to discover events or sets of events that match a predefined pattern of events describing a previously known attack. Misuse detectors can detect attacks quickly and reliably without high false positive rate. However, they can only detect attacks known for them, so they must be updated regularly with signatures of new attacks [7].

As you see, every technique has some strengths and weaknesses, so it is preferable to use misuse detection methods with anomaly detection components to have a more effective IDS. In this paper, we propose a hybrid model for host-based intrusion detection that uses both anomaly detection and misuse of detection techniques.

As mentioned in section 1, process mining is a way to extract knowledge from event logs produced by PAISs. Therefore, to develop HIDS in a PAIS, it is needed to use process mining techniques. In process mining techniques, three different perspectives are distinguished: (1) the process perspective, (2) organizational perspective, and (3) case perspective. The process perspective (also called control-flow perspective) focuses on control-flow i.e. the ordering of activities. The organizational perspective focuses on the originator field i.e. which users or roles are involved in, and how they are related. The case perspective focuses on properties of cases. Despite previous research works which only consider control-flow perspective, the proposed model considers both the control-flow and organizational perspective for detecting more attacks [8].

Four types of Host-based IDSs are defined that use different types of information: file system monitors, log file analyzers, connection analyzers, and kernel-based IDSs [9]. Since event logs are used as the starting point for process mining, the information needed for the proposed PAHID model is collected through log file monitoring.

Figure 1 provides an overview of the proposed Process Aware Host-based Intrusion Detection model. The model has three phases: preprocessing, hybrid anomaly detection and misuse detection, and merging results. In the following subsections, details of every phase and its implementation are described.

3.1 Preprocessing

As stated above, the starting point for process mining is event log. Throughout this paper, the term trace is used to refer to a process instance of a process model in the log and it represents the order in which activities are executed with all information recorded for every event. The goal of PAHID model is to discover attacked traces in the log.

The first step of the proposed PAHID model is related to keeping (or removing) some activities or traces from log that are (not) appropriate and important for analysis. Hence, three types of tasks can be executed in this step, based on the decision of the domain analyst:

- Removing incomplete traces. In every run of the PAHID model, the original log is related to a certain period, so there are traces that are not started or ended in this period. These incomplete traces are not fully recorded and are not started or ended with certain expected start and end events. Domain analyst can remove these traces before detection phase.
- Completing incomplete traces. Incomplete traces in the log can be the attacked traces by themselves, so removing all of them may cause some attacked traces not to be discovered. In order to fulfill this gap, domain analyst can complete these traces by adding artificial start and end events, if possible. Hence, these traces can be kept in the log and be analyzed.
- Removing irrelevant tasks/traces. Some tasks/traces in the process model/log are irrelevant or unimportant for analysis. Therefore, they can be removed to decrease the execution time and complexity of the detection phase.

For implementing this phase, ProM framework has some log filtering tools that can be used. To remove incomplete traces from log, **simple log filtering** tools can be used to define expected start and end activities. Then, traces that are not started and ended with these activities will be removed; also, **simple log filtering** tools can be used to define irrelevant tasks to be removed from traces. To keep incomplete traces and artificially complete them, **advanced log filtering** tools can be used to define artificial start and end tasks to be added to incomplete traces.

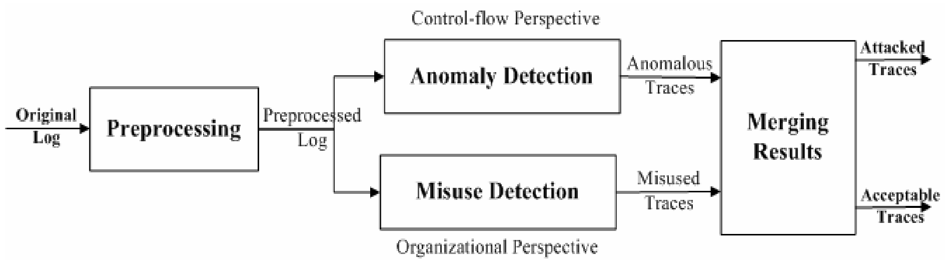

Figure 1. The Process Aware Host-based Intrusion Detection model

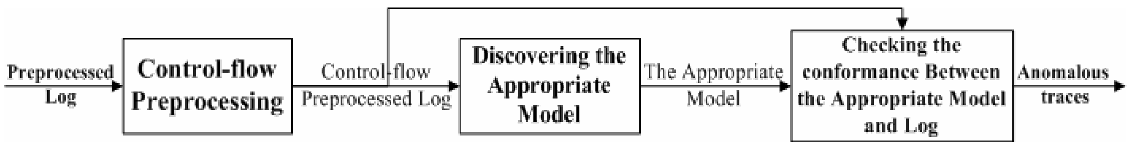
Preprocessed Log
Control-flow Preprocessing
Control-flow Preprocessed Log
Discovering the Appropriate Model
The Appropriate Model
Checking the conformance Between the Appropriate Model and Log
Anomalous traces

Figure 2. Steps of Anomaly Detection phase

3.2 Anomaly Detection

Anomaly detectors construct a reference model that represents normal behavior of the information system and users. Then, this reference model is used to analyze the current activities of the system looking for any deviation from it.

For constructing this reference model, an extensive training set (log) of normal behavior of the system and users is needed. Nevertheless, based on the unpredictable behavior of the users, this normal log cannot consider all of the possibilities or it is so complex. Moreover, in the flexible application domains, such a normal log is not known before execution of the information system.

Therefore, in the proposed model, it is assumed that a normal log/model is not available. The reference model is constructed during anomaly detection phase using current event log of the information system and has the most fitness with the behavior of the log in control-flow perspective. This reference model is constructed during anomaly detection phase using current reference model is called the appropriate model, M^*. Any deviation of the event log from the dynamically constructed appropriate model, M^*, is considered as an anomaly.

Figure 2 illustrates three steps of anomaly detection phase. Every step is described in the following.

3.2.1 Control-flow Preprocessing

To perform anomaly detection phase more quickly and easily, domain analyst can remove some events from log traces. These events are not related to control-flow perspective analysis. For example, tasks that will be analyzed in the misuse detection phase can be removed in this step.

Simple log filtering tools of ProM can be used to remove irrelevant events from traces in the log. The input is the preprocessed log, L^P, gained from the previous step and the output is the control-flow preprocessed log, L^{CP}, in MXML format.

3.2.2 Discovering the Appropriate Model

In order to classify normal and anomalous traces in a log, an appropriate process model should be discovered. This appropriate model should be complete and precise to present the behavior of the log correctly. Process mining, in control-flow perspective, can be considered as a search for the most appropriate model among the search space of candidate process models. Among different process mining techniques available for control-flow mining, genetic process mining is selected to discover the appropriate model in this work. Using genetic process mining has some advantages:

- In comparison with other process mining techniques, genetic process mining can deal with all the constructs possible in a log (sequences, parallelism, choices, loops, non-free-choices, invisible tasks and duplicate tasks).
- A system based on genetic algorithm can be retrained easily for changes. Therefore, the proposed PAHID model is flexible and it can respond rapidly to new market strategies and process models.
- Because of the parallelism in genetic algorithms, they can evaluate many process models at once; hence, they are suitable to solve problems with a large search space in a reasonable amount of time.

The genetic algorithm used in this step is supported by the **genetic algorithm** plug-in of the ProM framework. The control-flow preprocessed log, L^{CP}, gained from the first step, is used as the input for this step. The output is the most appropriate model, called M^*, discovered by genetic process mining that best describes the behavior of the control-flow preprocessed log in the form of **Heuristic net**.

For getting more information about the genetic process mining, the reader is referred to [10]–[12].

3.2.3 Checking the Conformance of the Appropriate Model and Log

In the last step of anomaly detection phase, it is time to classify the log to anomalous traces and normal traces. Therefore, conformance of every trace in the log is checked with the appropriate model. **Anomalous traces** are those in the log that do not fit the appropriate model.

For implementing this step, **conformance checker** plug-in of ProM can be used. A log and a defined process model are inputs of the **conformance checker** plug-in. This plug-in check the alignment between all traces in the log and the defined process model and discovers those traces that deviate from the process model. In this manner, traces in the log can be easily classified as follows: (1) fitting traces as normal traces, and then (2) other traces as anomalous traces.

The control-flow preprocessed log, L^{CP}, and the appropriate model, M^* (in the form of **Petri net)**, are inputs of the step. The outputs are anomalous traces in the log.

The input process model of **conformance checker** plug-in must be in the form of **Petri net**, but the output of the previous step (the appropriate model) is in the form of **Heuristic net**. Therefore, the appropriate model must be converted from **Heuristic net** to **Petri net** before using in **conformance checker** plug-in. **Conversion** plug-in of ProM is used for this purpose.

Summarizing, anomaly detection phase gets the preprocessed log, L^P, as input and discovers anomalous traces in control-flow perspective of the L^P as output.

Because of using anomaly detection technique, previously unknown attacks of the control-flow perspective can be discovered without any knowledge of details.

To get more information about anomaly detection phase, the reader is referred to [6].

3.3 Misuse Detection: Checking Attack rules

To increase the accuracy of detection, and to detect more attacks, misuse detection technique is used to detect misused traces in the organizational perspective.

Misuse detectors monitor the system activities to find predefined events or sets of events. These events or sets of events represent the behavior pattern of a known attack. These patterns can be defined in the form of rules as in this work. The rules are checked over all of the traces in the log. **Misused traces** are those in the log that fit any attack rule.

For implementing this step, four types of attacks related to organizational perspective are considered. These attacks can gain control of the information system by exploiting a variety of system flaws:

- User to Root (U2R): An authorized (legitimate) user gain unauthorized access to the information system.
- Remote to Root (R2R): An unauthorized remote user gain access to the information system from the Internet.
- Password Guessing: The intruder tries to guess the password of a user by entering incorrectly the username and password more than three times.
- Admin High Privilege Misuse: Administrator of the information system misuses his/her high privileges and performs activities of other users maliciously.

Four types of attacks are only considered for implementing the proposed model more quickly. Any other types and number of attacks can be implemented in the proposed PAHID model.

In order to develop attack rules, LTL[1] language is used. The LTL language exactly states properties of event logs of processes. A process can be considered as a list of process instances (traces). Every process instance itself is a list of M ordered audit trail entries that M can be different for any process instance. While checking LTL rules on a log, the log is checked one process instance at a time and every process instance is checked by inspecting its audit trail entries one by one. Using LTL language, every attack rule is developed in the form of formula. The formula can be developed in terms of propositional logic, quantifications, linear temporal logic, comparisons, or (sub) formula calls [13].

Four formulae are developed to represent behavior pattern of the attacks. **LTL Checker** analysis plug-in of ProM is used to check these rules over the preprocessed log. Inputs of this step are the preprocessed log, L^P, and an .ltl file containing developed attack rules. The outputs are misused traces fitting with any attack rule.

The rule developed for password guessing attack is shown below.

```
subformula login( u : ate.Originator ) :=
{}
    (  activity == "Login" /\ user == u  )
;

subformula guessing( u : ate.Originator ) :=
{}
   <> ( ( ( login(u)/\ _O ( login(u) ) )/\ _O
( _O ( login(u) ) ) ) )
;

formula Password_Guessing_Attack(  ) :=
{
}
   exists [ u : ate.Originator |  guessing( u
)  ]
;
```

This rule checks every process instance in order to find at least one user in it who tried to login more than three times. Such a process instance is a kind of misused trace.

3.4 Merging Results

In the previous hybrid phase, anomalies of control-flow perspective and misuses of organizational perspective are detected. Merging phase gets the result of the previous phase (anomalous traces and misused traces) and merges them as attacked traces with their attack types. Every attack can be an anomaly in activities order or any four types of attacks considered in misuse detection phase.

This phase is implemented through Merging program developed with Java programming language. The inputs are anomalous and misused traces and the outputs are attacked traces with their type of attacks.

Using the design descriptions stated above, Figure 3 illustrates the PAHID model with more details.

4. Experimental Results

We conducted an experimental evaluation on the proposed model to evaluate its efficiency. The evaluation is performed on a set of synthetic logs generated based on the log of a real-world information system.

Synthetic logs are used because of some reasons. First, since not all attacked traces are known completely and exactly even for security administrator in a real-world log, they cannot be used for evaluation. Second, real logs can be incomplete or contain noise and the last but not least; a real log is not available. Therefore, by using a synthetic log it is very easy to identify attacked traces and evaluate how well they are detected by the proposed model. Steps of performing the evaluation are described in the following subsections.

4.1 Generating Dataset

As mentioned in section 3, the information source for the PAHID model is log files produced by the host system. Two ways are possible for HIDSs to get their audit data: operating systems and application programs [14]. In this work, log files produced by the application program (PAIS) are used as an information source for the proposed model.

In order to generate a synthetic log related to an information system, application program of the Sara Tile Company is considered. The application program is installed on the host system of the company and is used by all of the computer nodes in the company network. Users can access the information system locally via the network or remotely via the Internet. Different types of processes are executed and all events executed by the users or information system are recorded on the host system of the network. Two types of processes performed in this company are considered for evaluation: purchase process and sale process. These two processes are executed for purchasing raw materials and selling company products. Every process has five different roles; every role contains one or more users.

[1] Linear Temporal Language

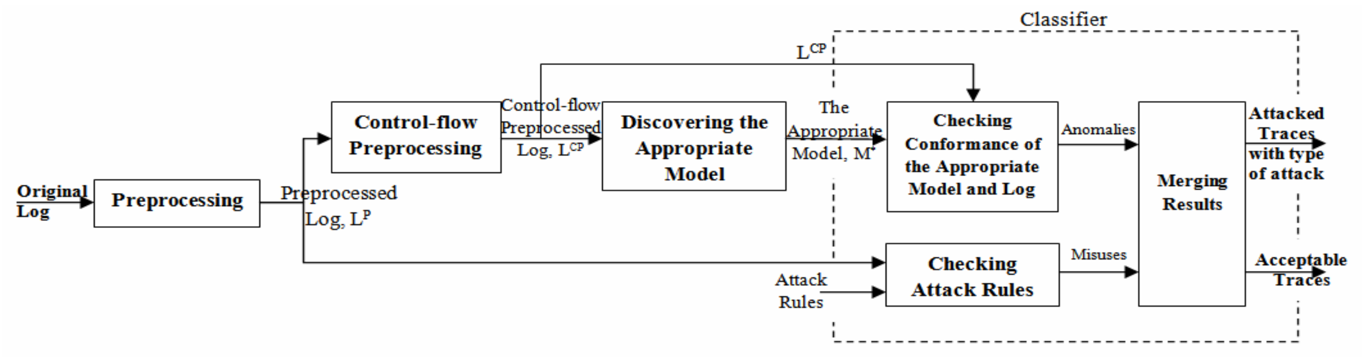
Classifier
L^{CP}
Original Log
Preprocessing
Preprocessed Log, L^P
Control-flow Preprocessing
Control-flow Preprocessed Log, L^{CP}
Discovering the Appropriate Model
The Appropriate Model, M
Checking Conformance of the Appropriate Model and Log
Anomalies
Attack Rules
Checking Attack Rules
Misuses
Merging Results
Attacked Traces with type of attack
Acceptable Traces

Figure 3. The PAHID model with more details

Table 1. Properties of training sets and test sets

	Test set of sale process	Training set of sale process	Test set of purchase process	Training set of purchase process
Average number of actions	452	4099	660	5963
Average number of users	10	21	10	12

CPN Tools are used to simulate the information system log. CPN Tools is a computer tool for constructing and analyzing CPN[2] models. A CPN model of a system is an executable model that represents the states of the system and events causing the system to change state. CPN models are organized as a set of modules using CPN language. CPN is a discrete-event modeling language that combines Petri nets with the functions of programming language standard ML. Petri nets provide basics for graphical notation and modeling concurrency, communication, and synchronization. Standard ML provides primitives for the definition of data types, describing data manipulation and creating compact and parametric models [15]. Two process models are modeled graphically in the CPN Tools as CP-nets. Every CP-net invokes the set of ML functions that will create event logs for every case executed. The models are simulated and every random run of the process (trace) is stored in a separate .cpnxml file (partial MXML logs).

ProM$_{\text{import}}$ framework is used to bundle these partial MXML logs into a single log that can be mined. **CPN Tools** plug-in of ProM$_{\text{import}}$ framework is used for this purpose. In this manner, two .mxml log files (dataset) for two processes are produced. The proposed model is evaluated two times for two processes to gain more accurate results.

4.2 Evaluation Method

To perform the evaluation, the dataset is partitioned to training and test sets. The appropriate model of preprocessed training set is discovered, and then conformance checking step, misuse detection, and merge phases are performed on the test set to evaluate the proposed model. For this purpose, **stratified 10-fold cross validation**[3] method is used. In this method, the dataset is partitioned to 10 equally folds, such that every fold is a good representative of the whole dataset. The model is executed 10 times and in every execution, a different fold of data is held-out for validation while the remaining nine folds are used for learning. The result (some predefined performance metrics) is average of the 10 results gained through ten execution of the model [16]. Properties of training sets and test sets generated for two processes are shown in Table 1.

Every dataset, generated in the previous step, have 500 instances (traces). Twenty percent of traces are attacked traces with different types of attacks (one to five numbers) inserted to the dataset. Datasets are partitioned to 10 folds of 50 traces; every fold has 10 attacked traces (20 %). The proposed model is executed 10 times for every dataset.

The results are given in the next subsection.

4.3 Results

For evaluating the proposed model, ROC[4] graph is used. ROC graphs are used for organizing classifiers and visualizing their performance to select the best one. In our model, we have a classifier that classifies traces of the preprocessed log to attacked traces and acceptable traces (see Figure 3).

In a classification problem, each instance in the data is mapped to one of the positive and negative classes. In our model, attacked traces are considered as the positive class and acceptable traces as the negative class. A classifier is a mapping from instances to predicted classes. Discrete classification models, such as our model, produce a discrete class label indicating only the predicted class of the instance. Given a classifier and an instance, there are

[2] Colored Petri Net

[3] This method is recommended as the best method for estimating the accuracy of classifiers [16].

[4] Receiver Operating Characteristics

four possible outcomes: a positive instance classified as positive is counted as true positive; if it is classified as negative, it is counted as false positive. A negative instance classified as negative is counted as true negative; if it is classified as negative, it is counted as false negative. Based on these outcomes, some metrics are defined to evaluate the performance of classifiers [17]. Equation (1) depicts the calculation formula of these metrics [17]:

$$fprate = \frac{FP}{N}, tprate = \frac{TP}{P}, Accuracy = \frac{TP+TN}{P+N} \quad (1)$$

The false positive rate (fp rate), is estimated by dividing the number of negatives incorrectly classified (fp) to total negatives (n). The true positive rate (tp rate) of a classifier is estimated by dividing the number of positives correctly classified (tp) to total positives (p). The accuracy is estimated by dividing the number of positives and negatives correctly classified (tp+tn) to total instances (p+n). Values of these metrics estimated for anomaly detection phase of the proposed model are shown in Table 2.

Table 2. Performance Metrics of Anomaly Detection phase

	Accuracy	TP rate	FP rate
Purchase Process	95 %	85 %	4.52 %
Sale Process	94.4 %	86.25 %	3.57 %

Since previous researches only have anomaly detection of control-flow perspective; the average result of this phase is compared with sampling method, introduced as the best method in [4]. Figure 4 depicts ROC graph utilized to compare the average performance of anomaly detection phase of the PAHID model with sampling method. Summary of results are shown in Table 3.

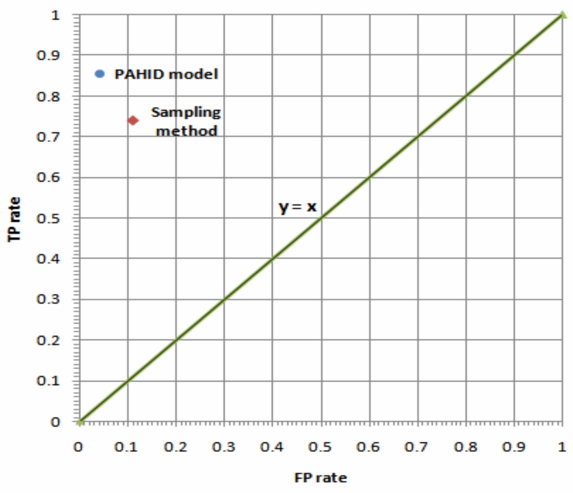

Figure 4. ROC graph of anomaly detection phase of PAHID model and Sampling method

Table 3. Performance Metrics of Anomaly Detection phase of PAHID model and Sampling method

	Accuracy	TP rate	FP rate
Anomaly Detection phase of PAHID model	94.7 %	85.62 %	4.04 %
Sampling method	85.96 %	74.04 %	11.01 %

ROC graph is a two-dimensional graph that fp rate is plotted on x-axis and tp rate on y-axis. A discrete classifier produces a (fp rate, tp rate) pair related to a single point in ROC space. One classifier point in the ROC space is better, if it is to the northwest of another point (higher fp rate or lower tp rate). Therefore, the point (0, 1) represents the perfect classifier. The diagonal line y=x represents the strategy of randomly guessing the classes of instances. Any classifier being in the lower right triangle is worse than random guessing and any classifier being in the upper left triangle is better than random guessing with correctly applying useful information [17].

Values of performance metrics estimated for the whole PAHID model are shown in Table 4.

Table 4. Performance Metrics of the Whole PAHID model

	Accuracy	TP rate	FP rate
Purchase Process	94.2 %	88 %	4.25 %
Sale Process	94.8 %	89 %	3.75 %

All results are discussed in the next subsection.

4.4Discussion

Followings are the results gained from evaluation of the PAHID model:

- Based on the average performance metrics illustrated in Table 3, the PAHID model in anomaly detection phase has better performance than sampling method. Since, the accuracy is increased nine percent, the true positive rate is increased 11.5 %, and the false positive rate is decreased seven percent.
- As depicted in Fig. 4, the PAHID model in anomaly detection phase is to the northwest of the sampling method, so it has better performance (higher accuracy and lower false positive rate) than sampling method. It means steps defined for anomaly detection phase are effective.
- As you see in Table 4. , the whole PAHID model has low false positive rate and high accuracy, so it has a good performance for detecting attacked traces in a PAIS. It means using anomaly detection and misuse detection in a hybrid manner is an effective way.
- Two process models (with all of the constructs possible in a log) are only considered for better evaluation. In fact, the proposed model does not need to know the process model before execution (adaptability). Also, any kind or number of processes is discoverable in practical application of the proposed PAHID model.

- Because of the parallelism in genetic algorithm, they can evaluate many process models at once. So, the proposed PAHID model can deal with a large search space in a reasonable amount of time when used in practical application.

5. Conclusion

In this paper, a Process Aware Host-based Intrusion Detection model is introduced. The model detects host-based intrusions in a PAIS using process mining in three phases: preprocessing, hybrid detection, and merging results. The model uses both anomaly detection and misuse detection techniques to provide more efficiency. Organizational perspective is also considered (rather than control-flow perspective) to detect more attacks. Because of using genetic process mining, the model is automated and flexible and can deal with large logs. The PAHID model is implemented by the use of ProM framework and Java programming. Evaluation results show that the proposed model provides high accuracy and low false positive rate.

The PAHID model considers control-flow and organizational perspectives. However, fraud may follow normal path in an authorized manner, but producing anomalous data (e.g. requesting a very large amount of fund). Therefore, in future works case perspective should also be considered to provide more accuracy.

In order to evaluate the efficiency of the proposed model better and have an experimental result with higher quality, it is recommended to perform an evaluation on a set of real log in the future.

References

[1] H. Debar, "an Introduction to Intrusion-Detection Systems", Proceedings of Connect'2000, Doha, Qatar, 2000.

[2] W.M.P. van der Aalst, A.K.A. de Medeiros, "Process mining and security: Detecting anomalous process executions and checking process conformance", Electronic Notes in Theoretical Computer Science, Vol 121, No. 4 , pp. 3–21, 2005.

[3] F. Bezerra, J. Wainer, "Anomaly detection algorithms in logs of process aware systems", SAC 2008: Proceedings of the 2008 ACM symposium on Applied computing, ACM Press, pp. 951–952, 2008.

[4] F. Bezerra, J. Wainer "Anomaly detection algorithms in business process logs", ICEIS 2008: Proceedings of the Tenth International Conference on Enterprise Information Systems, AIDSS, pp. 11–18, 2008.

[5] F. Bezerra, J. Wainer, W. van der Aalst "Anomaly detection using process mining", Springer-Verlag, Berlin Heidelberg, pp. 149–161, 2009.

[6] H. Jalali, A. Baraani, "Genetic-based Anomaly Detection in Logs of Process Aware Systems", WASET 2010: Proceedings of International Conference of Computer Science and Systems Security, pp. 251-256, 2010.

[7] R. Bace, P. Mell, "Intrusion Detection Systems", NIST Special Publication on Intrusion Detection System, 2004.

[8] W.M.P. van der Aalst, A.J.M.M. Weijters, "Process-Aware Information Systems: Bridging People and Software through Process Technology", Wiley & Sons, 2005.

[9] P. De Boer, M. Pels, "Host-based Intrusion Detection Systems", 2005.

[10] W.M.P. van der Aalst, A.K. Alves de Medeiros, A.J.M.M. Weijters, "Genetic process mining", Applications and theory of Petri nets, Springer, 2005.

[11] A.K.A. De Medeiros, A.J.M.M. Weijters, W.M.P. van der Aalst, "Using genetic algorithms to mine process models: Representation, operators and results", BETA Working Paper Series, WP 124, Eindhoven University of Technology, Eindhoven, 2004.

[12] A.K. Alves De Medeiros, "Genetic Process Mining", University Press Facilities, Eindhoven, 2006.

[13] HT. De Beer, PCW van den Brand, "The LTL Checker Plugins a (reference) manual", Eindhoven University of Technology, 2007.

[14] G. Vigna, C. Kruegel, "Handbook of Information Security", John Wiley and Sons, 2005.

[15] K. Jensen, L.M. Kristensen, L. Wells, "Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent System", International Journal on Software Tools for Technology Transfer (STTT), Vol. 9, No. 3-4, 2007.

[16] P. Rafeilzadeh, L. Tang, H. Liu, "Cross Validation", In Encyclopedia of Database Systems, Editors: M. Tamer Özsu and Ling Liu., Springer, 2009.

[17] T. Fawcett, "ROC Graphs: Notes and Practical Considerations for Researchers", Kluwer Academic Publishers, pp.1-38, 2004.