ITC 2/47
Journal of Information Technology and Control
Vol. 47 / No. 2 / 2018
pp. 220-235
DOI 10.5755/j01.itc.47.2.16277

Intrusion Detection in Cyber-Physical Systems Based on Petri Net

Received 2016/09/22 | Accepted after revision 2018/03/07

http://dx.doi.org/10.5755/j01.itc.47.2.16277

Intrusion Detection in Cyber-Physical Systems Based on Petri Net

Z. Ghazi, A. Doustmohammadi
Department of Electrical Engineering, Amirkabir University (Tehran Polytechnic),
e-mails: z.ghazi@aut.ac.ir, dad@aut.ac.ir

Corresponding author: dad@aut.ac.ir

Intrusion detection is a major concern in Cyber-Physical Systems (CPSs). In this paper, an algorithm based on Petri Net (PN) is proposed that simultaneously detects misuse and anomaly behavior of the system. The proposed anomaly detection method is applicable to Supervisory Control and Data Acquisition (SCADA) system at the highest level of CPSs. Neural First Order Hybrid Petri Net model (NFOHPN) with online fast Independent Component Analysis (ICA) is proposed for anomaly detection. It is shown that the use of distributed and multidisciplinary intrusion detection methods in different layers of CPSs increases security of the net against coordinated cyber-attacks. Simulation results and comparative studies based on the Defense Advanced Research Projects Agency (DARPA) evaluation datasets demonstrate that the proposed model can detect normal or malicious behavior with satisfying accuracy and at surprisingly high convergence speed.

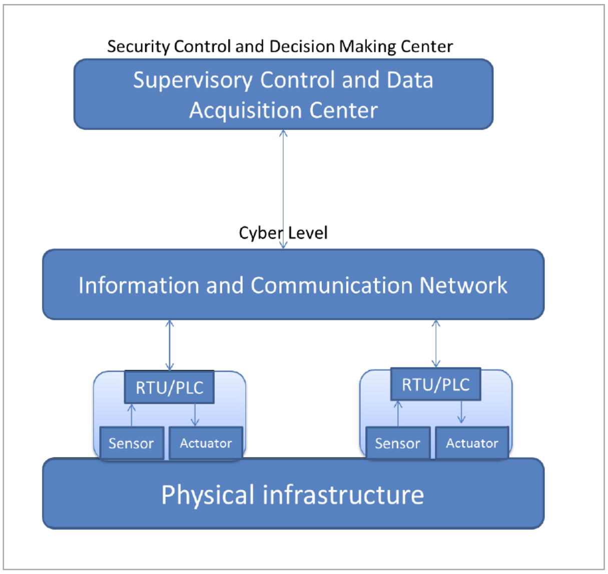
KEYWORDS: intrusion detection, Petri net, cyber-physical systems, neural network, independent component analysis.

1. Introduction

A cyber physical system (CPS) can be thought of a system that is comprised of sensors, actuators, and networking modules, which are applicable to areas such as energy, automotive, manufacturing, civil infrastructure, healthcare, and many others [31]. A high level view of CPSs is shown in Fig. 1 [34].

CPSs are complex systems where physical operations and cyber domain are supported coordinately. Even though Information and Communication Technology (ICT) has made a significant progress in CPSs, cyber security is still a big challenge in critical systems [36]. One of the most serious vulnerabilities in CPSs is intrusion attacks. Since early 21st century, significant attentions have been devoted to enhance security in CPSs [31, 15].

Figure 1
Architecture of cyber physical systems



Intrusion detection is a critical task for improving security in CPSs. To perform such a task, an intrusion detection system (IDS) is needed [30]. The idea of intrusion detection (ID) was first proposed by Anderson in 1980 [5]. Since then, plenty of research has been devoted on IDS [28, 33].

Intrusion detection schemes can be classified into two main classes of misuse and anomaly detection. In the first scenario, features of known attacks or system vulnerabilities are exploited for detection of misuses. When performing misuse detection, the audited data are compared with the database, and any compliance will be reported as an intrusion. Misuse detectors produce very few false positives; however, these types of detectors have their own shortcomings as well. For instance, creating and upgrading a comprehensive database is a cumbersome task. Moreover, they can only detect previously known attacks. Several techniques have been proposed for misuse detection.

Abbes et al. have proposed a novel protocol analysis approach to improve the performance of pattern matching [1]. Pattern matching based intrusion detection has been evaluated by Kreibich and Crowcroft [6]. Rule-based expert system is applied for misuse detection [4, 32]. Genetic algorithm is also used for misuse detection [13]. In recent years, data mining techniques have been applied to networks for building misuse detection models [24, 11, 27]. An overview of intrusion detection system using genetic algorithm and data mining is presented in [22]. Some efforts have been focused on classification and detection of computer intrusions using Colored Petri-nets [23, 12, 16].

Anomaly detectors pursue the normal behavior of the system. Any considerable deviation from usual operation of the system is labeled as an intrusion. The main advantage of these detectors is the ability to discover the attacks which are previously unknown. In contrast to the former, they generate many false positives, and hence, their accuracy is typically low. Many researches have been carried out by following this approach.

Some detection frameworks are based on clustering techniques [17, 35, 40]. In recent years, artificial leaning techniques have been widely used in anomaly detection. Some anomaly detection techniques in this category can be outlined as neural network [25, 8, 39], genetic algorithm [26, 37], and wavelet [3].

Early researches on intrusion detection systems consider both misuse detection as well as anomaly detection. As mentioned before, both misuse and anomaly detection systems have limitations and shortcomings. Most of existing intrusion detection structures can only identify either misuse or anomaly attacks. Simultaneous misuse and anomaly IDS have been proposed to overcome such shortages [21, 10].

Petri net is a powerful, fascinating, and graphical tool allowing to the operator to communicate effectively with the graphical programming environment. Theoretical calculations of PN are to a great extent simpler than other approaches leading to a much faster processing time which is so vital for IDS.

In this paper, PN is used for both misuse and anomaly detection in CPSs. Fundamentals of First Order Hybrid Petri Net (FOHPN) are described in Section 2. In Section 3, a PN model is proposed for misuse detection and its performance is validated. Neural network based FOHPN model along with experimental results are suggested in Section 4. Finally, conclusions are drawn in Section 5.

2. FOHPN Fundamentals

FOHPN is broad enough to model classes of systems of practical interest [7]. For a more comprehensive discussion on FOHPN, the reader is referred to [7] and the formulation and notation that follow in this section are taken from there.

An FOHPN model and its structure are denoted by $\langle N, m(\tau_0)\rangle$ and $N = (P, T, Pre, Post, D, C)$, where N and $m(\tau_0)$ represent FOHPN system and initial marking of the net respectively. P is the set of places that is partitioned into a set of discrete places P_d (represented as circles) and a set of continuous places P_c (represented as double circles), i.e. $P = P_d \cup P_c$, and T is the set of transitions that is partitioned into a set of discrete transitions T_d and a set of continuous transitions T_c (represented as double boxes), i.e. $T = T_c \cup T_d$. Furthermore, the set T_d is partitioned into a set of immediate transitions T_I (represented as bars), a set of deterministic timed transitions T_D (represented as black boxes), and a set of exponentially distributed timed transitions T_E (represented as white boxes), i.e. $T_d = T_I \cup T_D \cup T_E$. The cardinality of T, T_d, T_c is denoted by n, n_d, n_c, respectively.

The function $C : T_c \to R_0^+ \times R_\infty^+$ shows the firing speeds of continuous transitions. The pre- and post-incidence functions are as follows:

$$pre:\begin{cases} P_d \times T \to N \\ P_c \times T \to R_0^+ \end{cases} \qquad post:\begin{cases} P_d \times T \to N \\ P_c \times T \to R_0^+ . \end{cases} \tag{1}$$

for all $t \in T_c$ and $p \in P_d$, $Pre(p,t) = Post(p,t)$.

$D : T_d / T_I \to R^+$ shows the timing associated with the timed discrete transitions. A constant firing delay $\delta_i = D(t_i)$ is set for each deterministic timed transition $t_i \in T_D$. We assign an average firing rate $\lambda_i = D(t_i)$ for an exponentially distributed timed transition $t_i \in T_E$, and consequently, the average firing delay is $1/\lambda_i$, where λ_i is the parameter of the corresponding exponential distribution.

$C : T_c \to R_0^+ \times R_\infty^+$ defines the firing speeds related to continuous transitions, where $R_0^+ = R^+ \cup \{0\}$, $R_\infty^+ = R^+ \cup \{\infty\}$. For any continuous transition $t_i \in T_c$, let $C(t_i) = (V_i', V_i)$, with $V_i' \le V_i$. Here V_i' and V_i stand for the minimum firing speed (mfs), and the maximum firing speed (MFS), respectively.

The marking $m:\begin{cases} P_d \to N \\ P_c \to R_0^+ \end{cases}$ is a function that assigns to each discrete place a nonnegative number of tokens, and assigns to each continuous place a fluid volume; m_p denotes the marking of place p. The value of a marking at time τ is denoted by $m(\tau)$. The notations m^d and m^c show discrete and continuous marking, respectively.

Some advantages of using FOHPN are as follows:

- Petri net is a powerful and fascinating graphical tool and its application in modeling allows the operator to communicate effectively with the graphical programming environment and establishes a proper understanding of the model and the components of the system.
- Parallel, synchronous, and concurrent operations can be modeled fairly easy using PN.
- Mathematics and theoretical calculations of PN are to a great extent simpler than existing approaches and this usually leads to a much faster processing time that is so needed in IDS.
- CPSs have both continuous and discrete event signals and the interactions between these two cannot be ignored. HPN is an excellent tool for integrating these two dynamics together.

3. Misuse Detection in CPSs Based on Petri Net

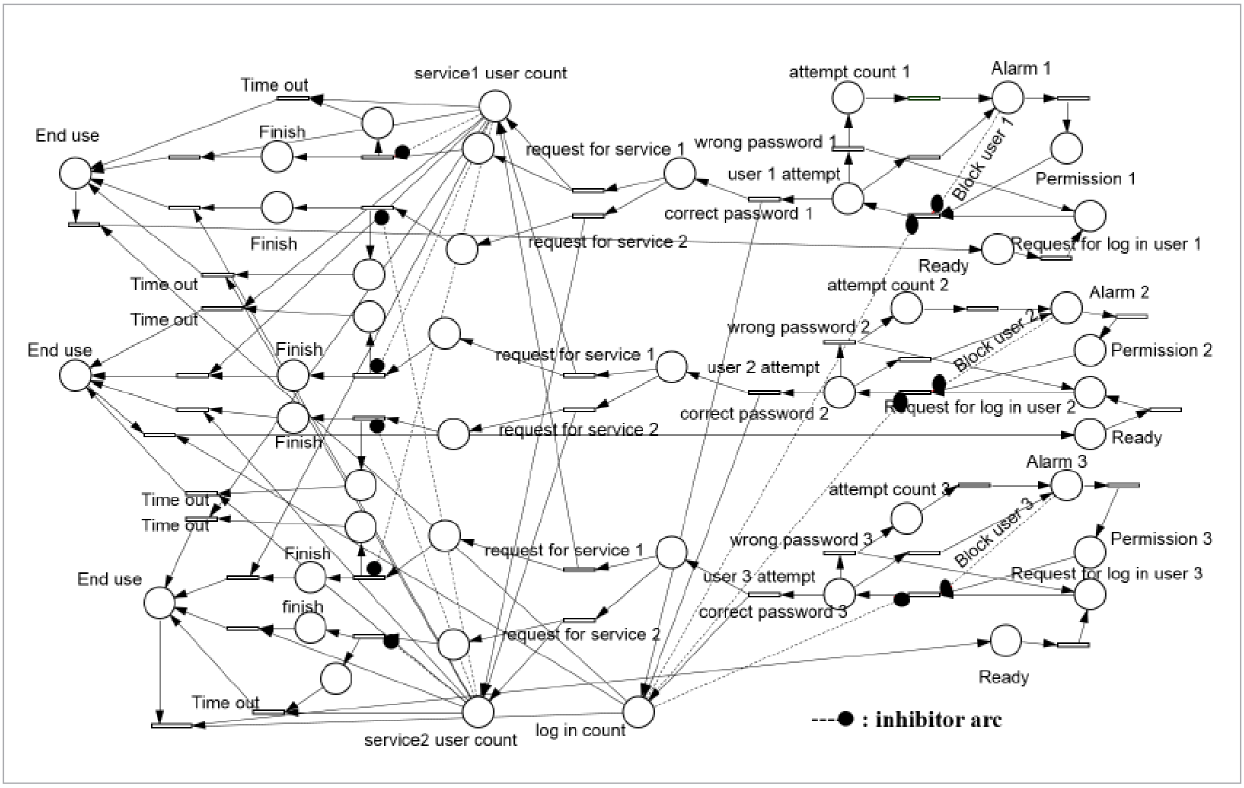
In this section, a PN model for misuse detection is proposed. Using simulations, efficiency of the proposed model and its capabilities are demonstrated. Fig. 2 shows the proposed PN model of IDS.

In the proposed model, three different types of users are defined: “normal user or user 1”, “emergency user or user 2” that is politically and socially sensitive, and an “unknown user or user 3”. This model has some specific capabilities that are listed below:

- For each user type, an allowable number of failed login attempts (num_failed_logins) can be assigned by supervisor. When the number of failed attempts reaches the pre-assigned number, then an alarm will be raised.
- For each user type, once the failed login attempts alarm is raised, the user must wait a certain amount of time before he/she is allowed to login.
- For each user type, the user must login within a certain amount of time after user name entry.

Figure 2

The proposed PN model for misuse detection



- Number of connections to a particular service (count number) is limited. It can be considered in the PN model. Priority for login can be set by supervisor for each user.
- Various rules can be set in the model. For example, in the proposed model, if emergency user is connected to a service to transmit crucial data, any other user can be blocked.
- To improve security of data transmission, an alarm is raised if data are not received after specific time interval. For example, 10 minutes.
- Maximum number of transferred data bytes between source and destination and vice versa (src or dst bytes) can be fixed for each user.
- Limited buffer space for sending or receiving can be considered in the proposed model.
- Supervisor can set allowable usage time for any specific service that may be different related to the users' needs.

Table 1

Imposed constraints in the proposed model

	Normal user	Emergency user	Unknown user
num_failed_logins	3	4	2
Password guessing time delay	3 min	5 min	2 min
allowable login time	1 hour	Not limited	30 min
Count	2		
Login priority	2	1	3
src bytes	100000	Not limited	50000
dst bytes	100000	Not limited	50000

Any constraint can be considered in PN model as well. For example, some constraints are listed in Table 1. They are considered in the proposed model (Fig. 2) as a sample and simulation results are presented later.

3.1 Simulation Results

The proposed model in Fig. 2 is simulated using MATLAB toolbox, and some results are presented in order to demonstrate the ability of PN in misuse detection. Violation from the defined constraints listed in Table 1, is considered as misuse, and the PN model can detect it as soon as possible. For instance, Fig. 3 shows the status of emergency user and unknown user. Whenever emergency user attempts to log in, the unknown user is blocked by supervisor in order to increase security.

Figure 3
The status of emergency user and unknown user

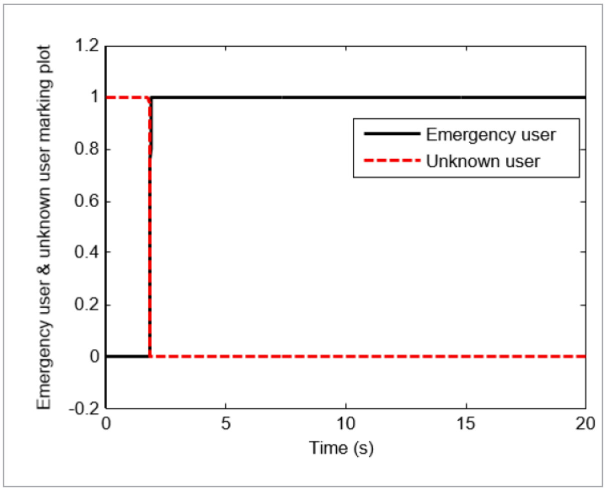


Fig. 4 illustrates the number of users that are connected to Service 1 at any given time. As can be seen from the figure, no more than two users are connected to Service 1 simultaneously and this is exactly what is expected based on the constraints that are imposed on the system. The jumps in Fig. 4 can be ignored and occur because of the delay between requesting from the third unpermitted user to be connected and firing of inhibitor arcs in order to block him.

Fig. 5 shows the number of times that a normal user is allowed to enter an incorrect password in a period of 180 seconds. As can be seen in the figure, the user is only allowed to attempt three incorrect passwords. In the proposed PN model (Fig. 2), if a normal user tries entering with wrong password more than three times that is shown in Fig.2 by firing of the transition "wrong password 1", the inhibitor arc "Block user 1", will fire and block the corresponding service for 10 minutes.

Figure 4
Number of users that are connected to Service1 at any given time

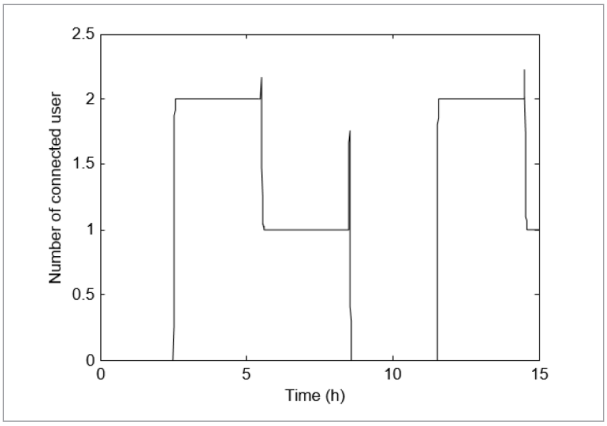
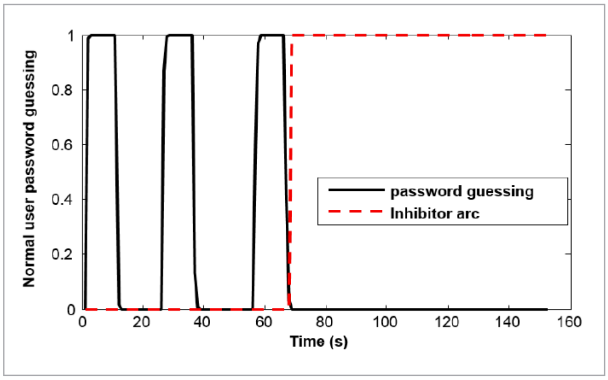


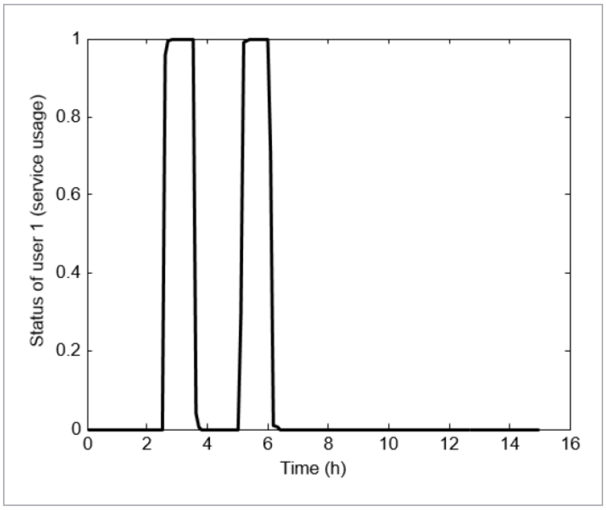
Figure 5
Password entry status by normal user



As was stated earlier, in the proposed model, one can limit the time interval that each user is allowed to use a specific service according to a predefined constraint. When this time interval expires, a transition that is shown with timeout in Fig. 2, will fire and the user will be disconnected from service. Fig. 6 shows the status of using the service by normal user. As can be seen, a normal user can only use the service for 1 hour continuously before he is disconnected from the network.

Simulation results (Figs. 4-6) illustrate the capability of the proposed PN model in misuse detection for CPSs. In the next section, neural FOHPN models are proposed for anomaly detection in SCADA system which is the center of security and decision making at the highest level of CPSs.

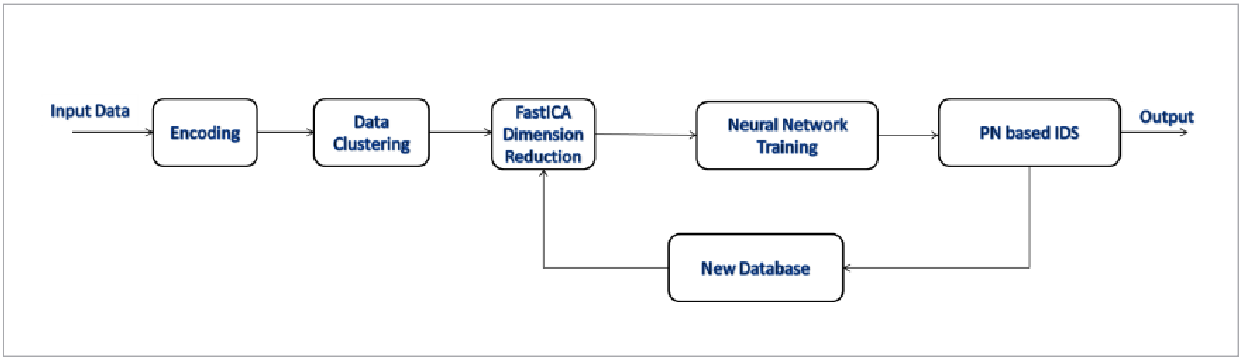
Figure 6
Status of normal user in terms of connecting to a specific service



4. Anomaly Detection in CPSs Based on Neural FOHPN

In this section, neural FOHPN is used for anomaly detection in CPSs. The 10% KDD 99 dataset is used for evaluating the proposed method. KDD training dataset consists of approximately 4,900,000 single connection vectors, which is so huge dataset for on-line detection. For order reduction and eliminating unnecessary records, first, KDD 99 dataset is encoded, and then the redundant records are omitted from the dataset. To extract independent features of such a large and complex dataset, FastAdaptiveOgICA [41] is implemented in order to reduce dimension of the dataset and make it more desirable for real time anomaly detection. Here, three neural FOHPN models are proposed for anomaly detection. The structure of the proposed approach is shown in Fig. 7.

Figure 7
The structure of the proposed anomaly detection approach



4.1. Feature Dataset

The KDD 99 dataset has around two million connection records in the two weeks of test. Each connection is labeled as either normal, or as an attack, with exactly one specific attack type, and contains 41 features (such as duration, protocol type, service, flag, source bytes, ...). Attacks fall into four main categories: DOS (denial of service), R2L (unauthorized access from a remote machine), U2R (unauthorized access to local superuser (root) privileges), and probing (surveillance and other probing) [18].

At the International Knowledge Discovery and Data Mining Tools Competition (IKDDMTC), "10% KDD 99" dataset is employed for the purpose of training. The attack categories along with the number of records that were attacked are listed in Table 2.

Based on [38], KDD 99 intrusion detection datasets suffer from large amount of redundant records in the train set. This causes the training operation to be time consuming and consequently ineligible for cyber security use. In the proposed intrusion detection algorithm, only distinct records in the KDD train set, listed in Table 3, are considered.

Table 2
10% KDD 99 intrusion detection dataset [18]

Category	Number of record	Atack	Number of record
DOS	391 458	Smurf	280790
		neptune	107201
		back	2203
		teardrop	979
		pod	264
		land	21
R21	1126	warezclient	1020
		guess_passwd	53
		warezmaster	20
		imap	12
		ftp_write	8
		multihop	7
		phf	4
		spy	2
U2R	52	buffer_overflow	30
		rootkit	10
		loadmodule	9
		perl	3
Probe	4107	satan	1589
		ipsweep	1247
		portsweep	1040
		nmap	231
Normal	97277		

Table 3
Number of distinct records in the KDD train set

Category	Normal	Probe	U2R	R2L	DOS
Distinct Train Records	2996	11656	52	995	45927
Distinct Test Records	9711	1106	37	2199	5741

4.2. Fast Independent Component Analysis (ICA)

Fast adaptive ICA algorithm is used in this paper in order to extract independent components of sampled data. ICA attempts to decompose a multivariate signal into independent non-Gaussian signals. ICA has been considered as a fundamental tool in the field of data analysis, feature extraction, and so on. The KDD 99 intrusion detection database includes non-Gaussian distribution, so ICA techniques are suitable for feature extraction in this field. Feature reduction plays an important role in case of improving ID performance and reducing the computational complexity, especially in an online detection.

The detailed formulation of FastAdaptiveOgICA is given in [41]. Not only is this method fast and adaptive with iterative neural network algorithm, it can also instantly separate mixture of sub-Gaussian and super-Gaussian signals. It has fast convergence speed and high separation performance.

Dimension reduction reduces the complexity of the proposed IDS significantly leading to less computational complexity and also fast and accurate intrusion detection. Implementation of fast adaptive ICA algorithm on the distinct KDD sample set leads to extraction of independent component of the dataset, and as a result, a new dataset with fewer dimensions is constructed. The new dataset is then used for neural network training in order to obtain Instantaneous Firing Speeds (IFSs) and the arc weights of the proposed FOHPN models. Table 4 shows the number of records before ICA and after implementing fast adaptive ICA algorithm.

Table 4
Number of data after implementing fast adaptive ICA algorithm in each category

Category	Normal	Probe	U2R	R2L	DOS
Dimension before ICA	2996	11656	52	995	45927
Reduced dimension	138	132	30	86	138

4.3. Proposed Neural FOHPN

Petri net has many similarities with neural networks in characteristics and rules [9]. Novel neural Petri net models have been considered by some researchers in the late 90's [9, 2]. A viable approach to detect cyber threats in CPSs appears to be neural FOHPN, since CPSs are event driven systems comprised of both discrete event and continuous dynamics. This is, in fact, the approach that is proposed in this paper.

4.3.1. Feedforward Neural FOHPN Architecture

For the purpose of anomaly detection, the FOHPN network shown in Fig. 8 consisting of the following three layers: input layer, rule layer, and output layer, are considered.

Define t_j as the set of transitions of the *jth* layer of the neural FOHPN, which the corresponding weights are assumed to be $w_1^j, w_2^j, \ldots, w_\eta^j$. In the sequel, first the formulation for the discrete transitions is approved. The net dynamic of FOHPN varies when a macro event occurs. In this situation, a discrete transition fires. This may change the discrete marking or enabling/disabling a continuous transition.

In the proposed model, when a discrete transition becomes firable, it just may change the marking of discrete transitions. Let $\sigma(\kappa)$ be the firing count vector at time instant κ. The macro-behavior of the FOHPN is defined during the κth macro-period by [7]:

$$\begin{aligned} m^d(\kappa+1) &= m^d(\kappa) + C_{dd}.\sigma(\kappa+1), \\ m^c(\kappa+1) &= m^c(\kappa). \end{aligned} \tag{2}$$

The output of the *jth* layer can be written as:

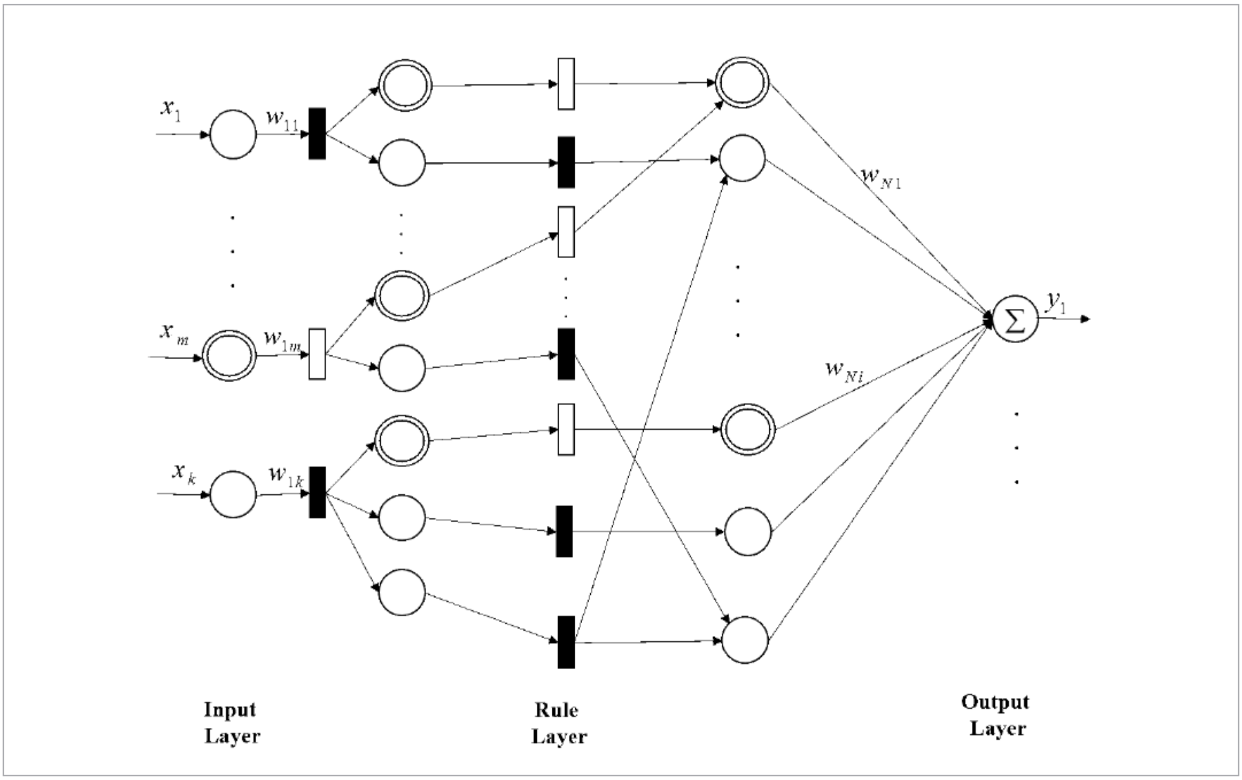
$$X^j = W^T M = \sum_{i=1}^{\eta} m(p_i^{\,j}) w_i^j,$$

$$Y(X^j) = F(X^j) = F(W^T M),$$

where m is the marking of the net as defined in (2). Now, the learning algorithm formulation in the rule layer is presented. Here, the back propagation method, which is the most common training method for

Figure 8

Framework of neural FOHPN



feedforward networks, is used. Useful information about neural network and recurrent neural network is invoked from [19].

We consider a neural FOHPN model having k rule layers. We therefore have a total of k + 2 layers including input and output layers that are numbered as $0, ..., k+1$.

The number of input and output places are r and L, respectively. There exist N places in the hidden layers. The training data for a feedforward network consist of q input-output data pairs. n denotes training instance. So, we define the input vector X° and the output vector Y as follows:

$$X^{\circ}(n) = [x_1^{\circ}(n), ..., x_k^{\circ}(n)]^T,$$
$$Y(n) = [y_1^{k+1}(n), ..., y_L^{k+1}(n)]^T,$$

Therefore, for the jth layer one can write:

$$\overline{y}_i^{j+1}(n) = F(\sum_{\beta=1,...,N^j} w_{i\beta}^j . x_\beta(n))$$
$$\overline{Y}(n) = [\overline{y}_1^{k+1}(n), ..., \overline{y}_L^{k+1}(n)]^T, \quad (3)$$

where $w_{i\beta}$ is the arc weight between i and β places in different layers. In order to find the arc weights of the network, an optimization problem must be solved. The following summed square error is considered as an objective function:

$$E = \sum_{n=1,...,q} \left\| Y(n) - \overline{Y}(n) \right\|^2 = \sum_{n=1,...,q} E(n). \quad (4)$$

Minimization of the cost function is done using gradient concept:

$$\frac{\partial E}{\partial w_{i\beta}^j} = \sum_{t=1,...,q} \frac{\partial E(n)}{\partial w_{i\beta}^j}.$$

By using a small learning rate γ, one can write:

$$w_{i\beta}^j(\omega+1) = w_{i\beta}^j(\omega) - \gamma \frac{\partial E(n)}{\partial w_{i\beta}^j}.$$

All training samples are used to update the new weights. One such pass through all samples is called an epoch. Arc weights should be initialized, typically to small random numbers, before the first epoch.

The gradient $\frac{\partial E(n)}{\partial w_{i\beta}^j}$ can be computed as follows:

$$\frac{\partial E(n)}{\partial w_{i\beta}^j} = \frac{\partial E(n)}{\partial Y(n)} * \frac{\partial Y(n)}{\partial w_{i\beta}^j}$$
$$= \sum_{n=1,...,q} \left\| Y(n) - \overline{Y}(n) \right\| * \frac{\partial F(X^j)}{\partial X^j} * \frac{\partial X^j}{\partial w_{i\beta}^j}.$$

To each discrete transitions $t_i \in T_D$ in FOHPN, we assign an average firing rate $\lambda_i = D(t_i)$ for an exponentially distributed timed transition $t_i \in T_E$, hence:

$$\frac{\partial F(X^j)}{\partial X^j} = \frac{\mu^j}{1+\exp(-b(X^j - \lambda^j))}$$
$$+ \frac{\mu^j X^j b * \exp(-b(X^j - \lambda^j))}{[1+\exp(-b(X^j - \lambda^j))]^2},$$
$$\frac{\partial X^j}{\partial w_{i\beta}^j} = m(p_i^j).$$

Afterward, the iterative Algorithm 1 is proposed to minimize the error using back propagation method in each epoch for the discrete net of the proposed neural FOHPN model:

Algorithm 1:

Step 0: set $\omega = 0$, $\rho = P$ *(*ρ *is a predetermined maximal number of epochs), and error =* ε.

Step 1: initialize weights to some small random numbers.

Step 2: for each sample n, compute intermediate layers outputs, except for input layer, from (3).

Step 3: compute error propagation term of the output layer

$\Delta_i^j(n)$ backward through $j = k+1, k, ..., 1$.

$$\Delta_i^{k+1}(n) = [y_i(n) - \overline{y}_i(n)] \left. \frac{\partial F}{\partial X} \right|_{X = z_i^{k+1}},$$

where $z_i^j(n) = \sum_{\alpha=1}^{N^{j-1}} x_\alpha^{j-1}(n) w_{i\alpha}^{j-1}$.

Step 4: compute error propagation term of the rule layers as follows:

$$\Delta_i^j(n) = \sum_{\varphi=1}^{N^{j+1}} \Delta_\varphi^{j+1} w_{\varphi i}^j \frac{\partial F}{\partial X}\bigg|_{X=z_i^m}.$$

Step 5: update the weights according to

$$w_{i\beta}^{j-1}(\omega+1) = w_{i\beta}^{j-1}(\omega) + \gamma \sum_{n=1}^{q} \Delta_i^j(n) x_\beta^{j-1}(n).$$

Step 6: after each epoch, error should be computed based on (4). If error >ε, or ω < P, then set ω = ω +1 and go to step 1; else go to step 7.

Step 7: End.

Subsequently, the same formulation for continuous transitions of the proposed neural FOHPN model will be derived. The whole framework of the new formulation is similar to previous one with the following differences.

The IFS of time transition $t_i \in T_c$ is denoted by $v_i(\tau)$. The marking of a place $p \in P_c$ is represented by

$$m(\tau + d\tau) = m(\tau) + w.v(\tau)d(\tau),$$
$$\frac{m(\tau + d\tau) - m(\tau)}{d(\tau)} = w.v(\tau).$$

It is assumed that at time τ, no discrete transition is fired and all speeds are continuous. So, we have:

$$\frac{dm(\tau)}{d\tau} = \sum_{t_i \in T_c} C(p, t_i).v_i(\tau). \tag{5}$$

A macro-event occurs by firing of a discrete transition or when a continuous place becomes empty. Firing of a discrete transition changes the discrete marking or enables/disables a continuous transition. The enabling state of a continuous transition changes from strong to weak, when a continuous place becomes empty. Let τ_k, τ_{k+1} be the occurrence times of macro-events; the time interval is called macro-period. Furthermore, it is assumed that during a macro-period the IFS of continuous transitions are constant, and τ_0 is the initial time, $\tau_k (k > 0)$ is the instants in which macro-events occur, and $v(\tau_k)$ is the IFS vector during the macro-period of length Δ_k. The macro-behavior of an FOHPN during the *kth* macro-period is defined as follows [7]:

$$\begin{aligned} m^c(\tau) &= m^c(\tau_k) + C_{cc}.v(\tau_k).(\tau - \tau_k), \\ m^d(\tau) &= m^d(\tau_k). \end{aligned} \tag{6}$$

In this case, the *jth* layer output of the FOHPN can be written as

$$Y(X^j) = F(X^j) = F(W^T M),$$

where M is continuous marking defined by (6), and $F= v(t)$. The rest of the formulation is the same as previously proposed algorithm for discrete net and can be derived directly.

A typical time delay neural FOHPN is implemented as a feedforward neural FOHPN. It has three layers as introduced: input, hidden and output layers. Just a set of delays are added to the inputs. Time delay can be easily implemented in the PN by considering delay in the firing time of the transitions. Time delay neural FOHPN is applied in order to detect anomaly and the performance comparison is provided in Section 4.4.

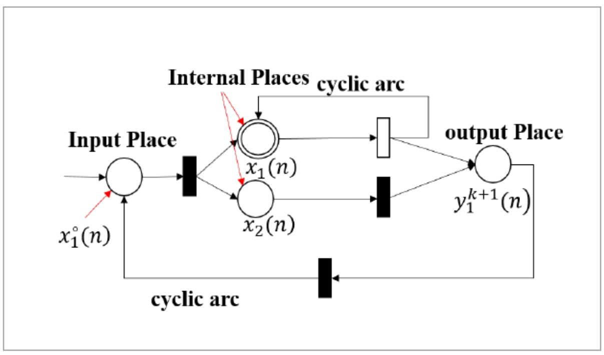
4.3.2. Recurrent Neural FOHPN Architecture

Recurrent neural network has lots of similarities with feedforward neural network, but in the former structure, at least one cyclic path exists. Since cyclic path can easily be incorporated by FOHPN model, the recurrent neural network structure can be applied in order to update the weights in the FOHPN. Fig. 9 shows a simple recurrent neural network FOHPN structure. This net with two cyclic arcs has one input place, two hidden places, and one output place.

Suppose that the recurrent neural FOHPN model has k input, N internal, and L output places represented by

Figure 9

An example of FOHPN with recurrent structure



$X^{\circ}(n)=[x_1^{\circ}(n),...,x_k^{\circ}(n)]^T$, $X(n)=[x_1(n),...,x_N(n)]^T$, $Y(n)=[y_1^{k+1}(n),...,y_L^{k+1}(n)]^T$, respectively. Furthermore, let $W^{in}, W, W^{out}, W^{back}$ be input $(N\times K)$, internal $(N\times N)$, output $(L\times(K+N))$, and back projection $(N\times L)$ connection weight matrices; respectively. F^{out} indicates a given activation function of the output layer. As a whole structure, we have:

$$X(n+1)=F(W^{in}X^{\circ}(n+1)+WX(n)+W^{back}Y(n))$$

$$Y(n+1)=F^{out}\left(W^{out}\left(X^{\circ}(n+1),X(n+1)\right)\right).$$

Discrete and continuous marking are the same as (1) and (5). Activation function F was defined for two cases of macro events occurrence. Back propagation algorithm is applied in the rule layer. The learning procedure is the same as previous one, but a new algorithm (based on [19]) must be proposed to iteratively update the weights:

Algorithm 2

Step 0: set $\omega = 0$, $\rho = P$ *(*ρ *is a predetermined maximal number of epochs), and* $error = \varepsilon$.

Step 1: first of all, weights should be initialized to small random numbers.

Step 2: for forward path, weights can be computed based on Algorithm 1.

Step 3: compute error propagation term of the output layer $\Delta_i(n)$ *by proceeding backward through* $n = q,...,1$, *for each sample n:*

$$\Delta_i(q)=[\sum_{j=1}^{L}[y_i(q)-\overline{y}_i(q)]\left.\frac{\partial F}{\partial X}\right|_{X=z_i(q)}$$

$$*w_{ji}^{out}]\left.\frac{\partial F}{\partial X}\right|_{X=z_i(n)}.$$

Step 4: compute error propagation term of the rule layers as follows:

$$\Delta_i(n)=[\sum_{\alpha=1}^{N}\Delta_\alpha(n+1)w_{\alpha i}$$

$$+\sum_{\alpha=1}^{L}\Delta_\alpha(n)w_{\alpha i}^{out}]\left.\frac{\partial F}{\partial X}\right|_{X=z_i(n)}.$$

where,

$$\Delta_\alpha(n)=[(y_\alpha(n)-\overline{y}_\alpha(n))$$

$$+\sum_{j=1}^{N}\Delta_j(n+1)w_{j\alpha}^{back}]\left.\frac{\partial F}{\partial X}\right|_{X=z_\alpha(n)}.$$

Step 5: update weights according to

$$w_{ij}^{in}(\omega+1)=w_{i\beta}^{j-1}(\omega)+\gamma\sum_{n=1}^{q}\Delta_i(n)x_j^{\circ}(n).$$

$$w_{ij}(\omega+1)=w_{ij}(\omega)+\gamma\sum_{n=1}^{q}\Delta_i(n)x_j(n-1).$$

$$w_{ij}^{out}(\omega+1)=w_{ij}^{out}(\omega)+\gamma\sum_{n=1}^{q}\Delta_i(n)x_j(n).$$

$$w_{ij}^{back}(\omega+1)=w_{ij}^{back}(\omega)+\gamma\sum_{n=1}^{q}\Delta_i(n)y_j(n-1).$$

Step 6: after each epoch, error should be computed based on (4) if If $error > \varepsilon$, *or* $\omega < P$ *or . Then, set* $\omega = \omega + 1$ *, and go to step 1;else go to step 7.*

Step 7: End.

4.4. Validation of the Approach and Simulation Results

Using the proposed feedforward neural FOHPN, time delay neural FOHPN, and recurrent neural FOHPN, three novel models were introduced for anomaly detection in CPSs. Measured features in KDD 99 dataset are defined as the initial conditions of the input places in the proposed models. A complete list of features defined for the connection records is given in Table 5.

As can be seen, some features of KDD 99 intrusion detection dataset are continuous and some are discrete. Consequently, the FOHPN is proposed in order to capture both feature types. All 41 features listed in Table 5 are used as the initial marking of the proposed neural FOHPN models. Based on the previously proposed training algorithms, appropriate unknown parameters such as arc weights, firing speeds, and time delays can be computed. After the training procedure, these parameters are set in the proposed models, and validation is done based on the test dataset.

In order to identify the normal behavior or cyber at-

Table 5
Basic features of KDD 99 intrusion detection dataset [18]

feature name	description	type
duration	length (number of seconds) of the connection	continuous
protocol_type	type of the protocol, e.g. tcp, udp, etc.	discrete
service	network service on the destination, e.g., http, telnet, etc.	discrete
src_bytes	number of data bytes from source to destination	continuous
dst_bytes	number of data bytes from destination to source	continuous
flag	normal or error status of the connection	discrete
land	1 if connection is from/to the same host/port; 0 otherwise	discrete
wrong_fragment	number of ``wrong'' fragments	continuous
urgent	number of urgent packets	continuous
hot	number of ``hot'' indicators	continuous
num_failed_logins	number of failed login attempts	continuous
logged_in	1 if successfully logged in; 0 otherwise	discrete
num_compromised	number of ``compromised'' conditions	continuous
root_shell	1 if root shell is obtained; 0 otherwise	discrete
su_attempted	1 if ``su root'' command attempted; 0 otherwise	discrete
num_root	number of ``root'' accesses	continuous
num_file_creations	number of file creation operations	continuous
num_shells	number of shell prompts	continuous
num_access_files	number of operations on access control files	continuous
num_outbound_cmds	number of outbound commands in an ftp session	continuous
is_hot_login	1 if the login belongs to the ``hot'' list; 0 otherwise	discrete
is_guest_login	1 if the login is a guest''login; 0 otherwise	discrete
count	number of connections to the same host as the current connection in the past two seconds	continuous
serror_rate	% of connections that have ``SYN'' errors	continuous
rerror_rate	% of connections that have ``REJ'' errors	continuous
same_srv_rate	% of connections to the same service	continuous
diff_srv_rate	% of connections to different services	continuous
srv_count	number of connections to the same service as the current connection in the past two seconds	continuous
srv_serror_rate	% of connections that have ``SYN'' errors	continuous
srv_rerror_rate	% of connections that have ``REJ'' errors	Continuous

tacks in the CPSs, three different methods, namely feedforward back propagation neural FOHPN, time delay neural FOHPN, and recurrent neural FOHPN have been used. Each model leads to a different accuracy and speed. Fig. 10 illustrates the performance of three different proposed FOHPN models. Performance comparison of these models, in terms of running time and accuracy, is provided in Table 6.

Figure 10

Performance evaluation of the (a) feedforward back propagation neural FOHPN, (b) recurrent neural FOHPN, (c) time delay neural FOHPN

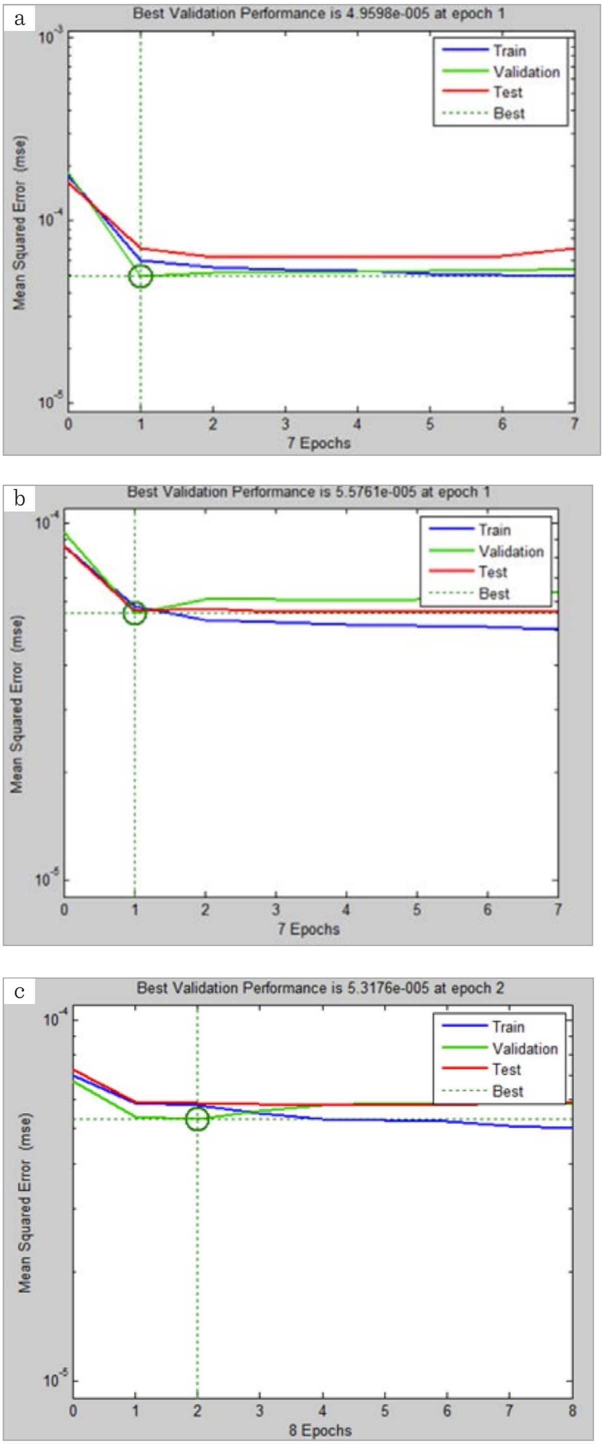


Table 6

Performance comparison of the proposed neural FOHPN models

	Running time (s)	Accuracy (norm of error)
Feedforward back propagation neural FOHPN	6	0.470
Recurrent neural FOHPN	149	0.224
time delay neural FOHPN	48	0.332

A comparison of detection rate (DR) for the DARPA test dataset using the proposed models is presented in Table 7.

Table 7

Detection rate comparison of the proposed FOHPN models

	Normal	Dos	Probe	U2R	R2L
Feedforward back propagation neural FOHPN	98.2	99.8	99.5	99.0	96.2
Recurrent neural FOHPN	97.9	99.4	98.7	98.5	94.0
Time delay neural FOHPN	99.2	98.5	99.2	98.9	96.0

Figure 11

Anomaly detection error based on feedforward neural FOHPN model

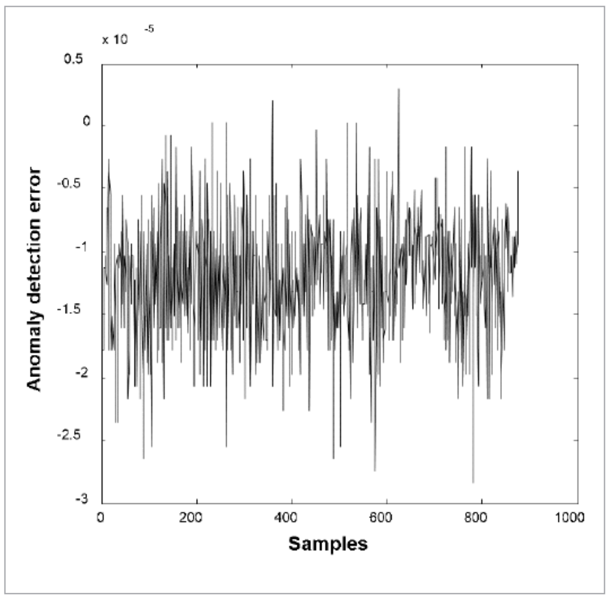


Table 8

Performance comparison of the proposed FOHPN model with other approaches

	Normal		Dos		Probe		U2R		R2L		Running
	DR	FPR	DR	FPR	DR	FPR	DR	FPR	DR	FPR	Time (s)
Feedforward back propagation neural FOHPN	98.2	2.9	100	1.6	99.5	1.2	99	0.6	96.2	0.4	0.1228
BPNN [14]	79.8	-	97.5	-	99.1	-	34.5	-	98.9	-	2.5
RBF [42]	-	-	98.8	1.6	98	1.6	-	-	97.2	1.6	1.6
HPCANN [29]	-	-	100	0.7	100	0.5	-	-	97.2	0.6	-
MLP [20]	--	--	99.9	--	48.1	--	48.3	--	93.2	--	30

Based on these results, all three models are accurate enough, but the running time of feedforward back propagation neural FOHPN is considerably smaller than others. Since the time of cyber detection is critically important, feedforward back propagation neural FOHPN seems to be more useful. Fig. 11 indicates anomaly detection errors based on feedforward neural FOHPN in all four attack categories and normal condition. As can be seen, the error on the test dataset is driven to a very small value.

To verify the effectiveness of the proposed model against existing neural-network-based approaches, a performance comparison of detection rate (DR) and false positive rate (FPR)[1] is provided in Table 8. All methods are implemented using MATLAB toolbox in the same PC.

Based on the result shown in Table 8, it is evident that the proposed feedforward back propagation neural FOHPN model is accurate enough to enhance acceptable detection rate. Its accuracy is better than [14], [42], and [20], and fairly close to [29], however, its running time is much smaller than the others. This factor is very critical in cyber intrusion detection concept. The speed of the proposed method is 13 times faster than the fastest one. As a result, one can conclude that not only feedforward back propagation neural FOHPN is satisfactorily accurate, but also the required time to detect abnormal or malicious behavior is much less than other methods. This efficiency is indeed expected based on Petri net properties.

1 It occurs when it is normal while IDS detects it attack.

5. Conclusion

In this paper, an intrusion detection method based on Petri net was proposed that was efficient in both real time misuse and anomaly detection. Petri net based misuse detection model was proposed in order to detect malicious behavior in physical or customer layer in CPSs. By implementing anomaly detection in SCADA layer of CPSs, one can monitor the overall systems and discover anomalous behavior. This phenomenon was implemented in this paper by using the proposed online neural FOHPN model. New extracted dataset was used in this paper based on DARPA evaluation dataset for implementing fast online intrusion detection. Simultaneous misuse and anomaly detection were proposed and implemented based on Petri net in CPSs. Furthermore, the convergence speed of the proposed model makes it suitable specially in intrusion detection of sensitive CPSs. Simulations were carried out to demonstrate the performance of the proposed model on DARPA evaluation datasets. The results illustrate significant improvement in speed and also a satisfactory accuracy compared to other existing intrusion detection approaches.

References

1. Abbes, T., Bouhoula, A. Rusinowitch, M. Protocol Analysis in Intrusion Detection Using Decision Tree. International Conference on Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004, 2004, 1, 404-408. https://doi.org/10.1109/ITCC.2004.1286488
2. Ahson, S. I. Petri Net Models of Fuzzy Neural Networks. IEEE Transactions on Systems, Man, and Cybernetics, 1995, 25(6), 926-932. https://doi.org/10.1109/21.384255
3. Alarcon-Aquino, V., Ramirez-Cortes, J. M., Gomez-Gil, P., Starostenko, O. Garcia-Gonzalez, Y. Network Intrusion Detection Using Self-Recurrent Wavelet Neural Network with Multidimensional Radial Wavelons. Information Technology and Control, 2014, 43(4), 347-358. https://doi.org/10.5755/j01.itc.43.4.4626
4. Anderson, D., Frivold, T. Valdes, A. Next-Generation Intrusion Detection Expert System (Nides): A Summary, 1995.
5. Anderson, J. P. Computer Security Threat Monitoring and Surveillance. Technical Report, James P. Anderson Company, 1980.
6. Antonatos, S., Anagnostakis, K. G. Markatos, E. P. Generating Realistic Workloads for Network Intrusion Detection Systems. ACM SIGSOFT Software Engineering Notes, ACM, 2004, 29, 207-215. https://doi.org/10.1145/974043.974078
7. Balduzzi, F., Giua, A. Menga, G. First-Order Hybrid Petri Nets: A Model for Optimization and Control. IEEE Transactions on Robotics and Automation, 2000, 16(4), 382-399. https://doi.org/10.1109/70.864231
8. Bitter, C., North, J., Elizondo, D. A., Watson, T. An Introduction to the Use of Neural Networks for Network Intrusion Detection. Computational Intelligence for Privacy and Security, Springer, 2012, 5-24.
9. Chow, T. W., Li, J.-Y. Higher-Order Petri Net Models Based on Artificial Neural Networks. Artificial Intelligence, 1997, 92(1-2), 289-300. https://doi.org/10.1016/S0004-3702(96)00048-3
10. Chung, Y. Y., Wahid, N. A Hybrid Network Intrusion Detection System Using Simplified Swarm Optimization (SSO). Applied Soft Computing, 2012, 12(9), 3014-3022. https://doi.org/10.1016/j.asoc.2012.04.020
11. Denatious, D. K., John, A. Survey on Data Mining Techniques to Enhance Intrusion Detection. IEEE International Conference on Computer Communication and Informatics (ICCCI), 2012, 1-5. https://doi.org/10.1109/ICCCI.2012.6158822
12. Dolgikh, A., Nykodym, T., Skormin, V., Antonakos, J., Baimukhamedov, M. Colored Petri Nets as the Enabling Technology in Intrusion Detection Systems. IEEE Military Communications Conference MILCOM, 2011, 1297-1301. https://doi.org/10.1109/MILCOM.2011.6127481
13. Goyal, M. K., Aggarwal, A. Composing Signatures for Misuse Intrusion Detection System Using Genetic Algorithm in an Offline Environment. Advances in Computing and Information Technology, Springer, 2012, 151-157.
14. Haddadi, F., Khanchi, S., Shetabi, M., Derhami, V. Intrusion Detection and Attack Classification Using Feed-Forward Neural Network. IEEE Second International Conference on Computer and Network Technology (ICCNT), 2010, 262-266. https://doi.org/10.1109/ICCNT.2010.28
15. Han, S., Xie, M., Chen, H.-H., Ling, Y. Intrusion Detection in Cyber-Physical Systems: Techniques and Challenges. IEEE Systems Journal, 2014, 8(4), 1052-1062. https://doi.org/10.1109/JSYST.2013.2257594
16. Helmer, G., Wong, J., Slagell, M., Honavar, V., Miller, L., Wang, Y., Wang, X., Stakhanova, N. Software Fault Tree and Coloured Petri Net–Based Specification, Design and Implementation of Agent-Based Intrusion Detection Systems. International Journal of Information and Computer Security, 2007, 1(1-2), 109-142. https://doi.org/10.1504/IJICS.2007.012246
17. Horng, S. J., Su, M. Y., Chen, Y. H., Kao, T. W., Chen, R. J., Lai, J. L., Perkasa, C. D. A Novel Intrusion Detection System Based on Hierarchical Clustering and Support Vector Machines. Expert Systems with Applications, 2011, 38(1), 306-313. https://doi.org/10.1016/j.eswa.2010.06.066
18. KDD Cup 1999 Data. Available at http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.
19. Jaeger, H. Tutorial on Training Recurrent Neural Networks, Covering BPPT, RTRL, EKF and the" Echo State Network" Approach, 2002, 5.
20. Jawhar, M. M. T., Mehrotra, M. Design Network Intrusion Detection System Using Hybrid Fuzzy-Neural Network. International Journal of Computer Science and Security, 2010, 4(3), 285-294.

21. Kim, G., Lee, S., Kim, S. A Novel Hybrid Intrusion Detection Method Integrating Anomaly Detection with Misuse Detection. Expert Systems with Applications, 2014, 41(4), 1690-1700. https://doi.org/10.1016/j.eswa.2013.08.066
22. Kshirsagar, V. K., Tidke, S. M., Vishnu, S. Intrusion Detection System Using Genetic Algorithm and Data Mining: An Overview. International Journal of Computer Science and Informatics ISSN (PRINT), 2012, 2231, 5292.
23. Kumar, S. Classification and Detection of Computer Intrusions. Ph.D. thesis, Purdue University, 1995.
24. Lee, W., Stolfo, S. J., Mok, K. W. Mining Audit Data to Build Intrusion Detection Models. KDD, 1998, 66-72.
25. Lei, J. Z., Ghorbani, A. A. Improved Competitive Learning Neural Networks for Network Intrusion and Fraud Detection. Neurocomputing, 2012, 75(1), 135-145. https://doi.org/10.1016/j.neucom.2011.02.021
26. Li, W. Using Genetic Algorithm for Network Intrusion Detection. Proceedings of the United States Department of Energy Cyber Security Group, 2004, 1, 1-8.
27. Li, Y., Wang, Y. A Misuse Intrusion Detection Model Based on Hybrid Classifier Algorithm. International Journal of Digital Content Technology and Its Applications, 2012, 6(5), 352-365.
28. Liao, H.-J., Lin, C.-H. R., Lin, Y.-C., Tung, K.-Y. Intrusion detection System: A Comprehensive Review. Journal of Network and Computer Applications, 2013, 36(1), 16-24. https://doi.org/10.1016/j.jnca.2012.09.004
29. Liu, G., Yi, Z., Yang, S. A Hierarchical Intrusion Detection Model Based on the PCA Neural Networks. Neurocomputing, 2007, 70(7-9), 1561-1568. https://doi.org/10.1016/j.neucom.2006.10.146
30. McHugh, J. Intrusion and Intrusion Detection. International Journal of Information Security, 2001, 1(1), 14-35. https://doi.org/10.1007/s102070100001
31. Mitchell, R., Chen, I.-R. A Survey of Intrusion Detection Techniques for Cyber-Physical Systems. ACM Computing Surveys (CSUR), 2014, 46(4), 55(1-29).
32. Mitchell, R., Chen, R. Behavior-Rule Based Intrusion Detection Systems for Safety Critical Smart Grid Applications. IEEE Transactions on Smart Grid, 2013, 4(3), 1254-1263. https://doi.org/10.1109/TSG.2013.2258948
33. Modi, C., Patel, D., Borisaniya, B., Patel, H., Patel, A., Rajarajan, M. A Survey of Intrusion Detection Techniques in Cloud. Journal of Network and Computer Applications, 2013, 36(1), 42-57. https://doi.org/10.1016/j.jnca.2012.05.003
34. Morris, T. H., Srivastava, A. K., Reaves, B., Pavurapu, K., Abdelwahed, S., Vaughn, R., McGrew, W., Dandass, Y. Engineering Future Cyber-Physical Energy Systems: Challenges, Research Needs, and Roadmap. IEEE North American Power Symposium (NAPS), 2009, 1-6. https://doi.org/10.1109/NAPS.2009.5484019
35. Muda, Z., Yassin, W., Sulaiman, M., Udzir, N. Intrusion Detection Based on K-Means Clustering and Naive Bayes Classification. IEEE 7th International Conference on Information Technology in Asia (CITA 11), 2011, 1-6.
36. Raciti, M. Anomaly Detection and Its Adaptation: Studies on Cyber-Physical Systems. Ph.D. thesis, Linköping University Electronic Press, 2013.
37. Srinivasu, P., Avadhani, P. Genetic Algorithm Based Weight Extraction Algorithm for Artificial Neural Network Classifier in Intrusion Detection. Procedia Engineering, 2012, 38, 144-153. https://doi.org/10.1016/j.proeng.2012.06.021
38. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A. A. A Detailed Analysis of the KDD Cup 99 Data Set. IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009, 1-6. https://doi.org/10.1109/CISDA.2009.5356528
39. Wang, G., Hao, J., Ma, J., Huang, L. A New Approach to intrusion Detection Using Artificial Neural Networks and Fuzzy Clustering. Expert Systems with Applications, 2010, 37(9), 6225-6232. https://doi.org/10.1016/j.eswa.2010.02.102
40. Xu, Y.-Q., Zhang, B., Qin, X.-T. Clustering Intrusion Detection Model Based on Grey Fuzzy K-Mean Clustering. Journal of Chongqing Normal University (Natural Science), 2013, 1, 0-19.
41. Ye, Y., Zhang, Z.-L., Zeng, J., Peng, L. A Fast and Adaptive ICA Algorithm with Its Application to Fetal Electrocardiogram Extraction. Applied Mathematics and Computation, 2008, 205(2), 799-806. https://doi.org/10.1016/j.amc.2008.05.117
42. Zhang, C., Jiang, J., Kamel, M. Intrusion Detection Using Hierarchical Neural Networks. Pattern Recognition Letters, 2005, 26(6), 779-791. https://doi.org/10.1016/j.patrec.2004.09.045