3

Particle Filter for Depth Evaluation of Networking Intrusion Detection Using Coloured Petri Nets

Chien-Chuan Lin and Ming-Shi Wang
Department of Engineering Science, National Cheng Kung University
Taiwan

1. Introduction

In this chapter, we investigated and proposed an approach that used the particle filter concept in a network intrusion detection system and simulated in the Coloured Petri Nets tools (CPN Tools) platform to trace and pre-detect networking attack and intrusion behaviors. We call this proposed approach as Network Particle Filter (NPF) scheme. We can realize what it happened by analyzing and simulating an intrusion in detail. The experimental results demonstrated that the Coloured Stochastic Petri Nets (CSPN) model approach is an efficient and helpful to evaluate an intrusion detection system in depth.

The motivation of this investigation is to consider that almost networking behavior could be marked when it has been done in a network. We are interesting to and focusing on trace and analyze the steps of networking intrusion behavior.

Any single attack or incident can be decomposed into multiple steps of intrusion behavior is called multistage attack. We can set up the defense scenarios by analyzing and combining those multi-stages. The most abnormal incidents may not be detected at their initial steps, but once their signatures are caught , the attack behaviors would also be known. On the other hand, the first steps of abnormal incidents look like normal events. The great damage has been done when they are been detected. To consider the advantages when those intrusion incidents can be pre-detected at, they did not make any damage yet. We list the following four advantages: (1) to reduce the attacked risk, (2) to reduce detect cost, (3) to increase system security and (4) to reach pre-detection.

Dynamic detecting for an intrusion detection system is also provided in this investigation. In traditional IDSs, the novel attacks and intrusion are impossible to detect and prevent. However, how to estimate the cost and risk to prevent attack or intrusion network systems are critical issues for network systems manager. According to flaw hypothesis methodology and Coloured Stochastic Petri Nets modeling, each attack model is given a corresponded with threshold. Monitoring and tracing behavior of attacks concurrently. The most attacks, however, can be detected or prevented by simulating and analyzing their incident types in depth.

The particle filter, also known as Sequential Monte Carlo method or said Condensation (Isard and Blake 1998), it is the use of the concept that the probability of particle sets to be used in any form of state space. Its core idea is likely to engage in the posterior to express its

distribution. In brief, the particle filter is a means to find a group in the state space of the random sample spread teams to approximate the probability density function to replace the sample mean points operations, thereby gaining the status of the process of distribution of minimum variance. As the number of samples is near infinite, the particle filter scheme can approach any form of probability density function.
The Kalman Filter, however, is based on the assumption target is the linear-type and the Gaussian distribution. The particle filter can be used in non-linear and non-Gaussian distribution model. The Particle filter can obtain a high detection accuracy and target trace rate which the main reason is that it can track the status of a random number of assumptions made at the same time retaining the possibility of a higher number of assumptions, not only left the state of a forecast. Therefore, when the target state of a sudden change in a matter of time before the prediction is wrong, the other particles can state the possibility of a higher state to amend the error.
Kristensen et al (Kristensen, Jorgensen et al. 2004) presented four case studies where CP-nets and their supporting computer tools are used in system development projects with industrial partners. The case studies have been selected such that they illustrate different application areas of CP-nets in various phases of system development. Kristensen and Jensen (Kristensen and Jensen 2004) presented two case studies where CP-nets and their supporting computer tools are used for ad-hoc networks. Dahl (Dahl 2005) and Dahl and Wolthusen (Dahl and Wolthusen 2006) addressed the flaw hypothesis methodology (FHM) to work at the intrusion detection system.
IP trace back is another issue for attack detection and analysis. In our investigation, IP trace back technologies are helpful to analyze and evaluate intrusion detection. Savage et al (Savage, Wetherall et al. 2001) described that trace back is only effective at finding the source of an attack traffic, not necessarily the attacker themselves. Savage et al also defined some basic assumptions and limitations for traffic trace back those are as follows. An attacker may generate any packet, multiple attackers may conspire, attackers may be aware they are being traced, packets may be lost or reordered, attackers send numerous packets, the route between attacker and victim is fairly stable, routers are both CPU and memory limited and routers are not widely compromised.
Snoeren (Snoeren, Partridge et al. 2002) gave another several important assumptions that a trace back system should make about a network and the traffic it carries. The packets may be addressed to more than one physical host, duplicate packets may exist in the network, routers may be subverted, but not often, attackers are aware they are being traced, the routing behavior of the network may be unstable, the packet size should not grow as a result of tracing and hosts may be resource constrained.
Steffan and Schumacher (Steffan and Schumacher 2002) presented the fault tree analysis (FTA) scheme, which fault tree technologies have been used to analyze the failure conditions of complex technical systems for a long time. Attack tree methods can capture the steps of an attack and their interdependencies. Attack tree methods are also used to represent and calculate probabilities, risks, cost, or other weightings. The main building blocks of attack trees are nodes. Each fault tree has a single top node which represents the achievement of the attack's ultimate goal. Interdependencies of goals are modeled by the tree hierarchy. Attack steps that have to be performed successfully before another step can occur are represented by child nodes. To each node either a logical AND or a logical OR gate is associated. An OR-node can occur when any of its child events occurs. For an AND-node

to occur its entire child events are necessary. Fault Tree nodes can be augmented with probabilities or costs, so that the most likely or inexpensive attack path can be calculated. However, those weightings are too specific to be applied to attack trees describing general attack scenarios.

Gordon (Gordon, Salmond et al. 1993) first proposed an algorithm of particle filters, known as a sequential importance resampling (SIR) filter. A key issue in SIR is the selection of the proposal distribution, which determines the approximation performance. Much research ofthe particle filtering focuses on improving the proposal distribution and importance sampling strategies by utilizing the measurements, such as the auxiliary particle filter (Pitt and Shephard 1999). Recently, some kernel based particle filters have been introduced, including Gaussian sum particle filter (Kotecha and Djuric 2003), kernel particle filter (Hurzeler and Kunsch 1998) and Parzen particle filter (Lehn-Schioler, Erdogmus et al. 2004), which enhance the ability of the particles in the posterior distribution representation by the kernel density estimators.

In a traditional particle filter scheme almost applied into trace visual object. In this research, we extend the particle filter function to analyze the network flows and evaluate the risk and cost of intrusion detection system work.

2. Background

2.1 Intrusion Detection System

Intrusion detection systems (IDS) detect attempted or successful misuses of computer systems. IDS can be classified according to their (1) data sources: network or host audit trails; (2) analysis technique: misuse or anomaly detection; and (3) overall architecture: distributed or autonomousagents.

The Host-based audit trails application and system logs, file attributes, system call and process monitoring, kernel audit facilities. Its problems are as follows. (1) It can't trust audit trail from a compromised host; (2) there is performance impact of active monitoring on target systems.

The Network-based audit trails raw packet data, network flow, and firewall and router logs. Its problems are as follows. (1) The passive network monitoring is easily defeated by clever attackers; (2) the traffic normalizer can help deal with ambiguity; (3) they require the higher bandwidth, end-to-end encryption and switched networks.

The misuse detection looks for specific, identifiable attacks, for example, expert knowledge IDS is rules-based according to attack signatures. Its problems are as follows. (1) It cannot detect novel attacks and (2) it is extremely brittle in the face of mutating attacks or subterfuge.

The Anomaly detection looks for anything that doesn't fit a normal profile. Those methods include following. (1) Equality matching that is a simple anomaly detection - detect deviance from specified normal behavior. Its main problems are an inability to generalize from past observed behavior and subject to state-holding or other denial of service attacks. (2) Statistical profiling that comprise profiles of normal behavior from various statistical measures. Its problems are insensitive to an event ordering and the threshold determination. (3) Machine learning that applies AI techniques (Elman, Petri, neural nets, etc.) to learn normal profiles. Its problems include those are extremely high false positives due to high sensitivity to variance, subject to bad training, and poor real-time performance, questionable real-world applicability.

In popularly, host IDS (HIDS) and network IDS (NIDS) are two kind IDSs. HIDS is to detect the possible intrusion and attack on a host by reviewing the audited data of the host. NIDS is to detect the possible intrusion and attack on a LAN by checking each networking packet on the LAN. The features matching scheme is the main technology for IDS. Although IDS can detect intrusion and attacks, but if the feature data were not been updated in time, then the detection rate would be decreased. Due to the IDS does not find out any new attack or intrusion behavior.

2.2 Coloured Petri Nets

Coloured Stochastic Petri Nets are now in widespread use for many different practical purposes (Jensen 1992). The main reason for the great success of these kinds of net models is the fact that they have a graphical representation and a well-defined semantics allowing formal analysis. Real-world systems often contain many parts, which are similar, but not identical. Using CSPN, these parts must be represented by disjoint sub nets with a nearly identical structure. The practical usages of CSPN to describe real-world systems have clearly demonstrated a need for more powerful net types, to describe complex systems in a manageable way.
The formal definition of a Petri Net graph is as follows (Dahl 2005): A *Petri net graph* G is a bipartite directed multigraph, G = (V, A), where V = v_1, v_2, v_3, …, v_n is a set of vertices and A = a_1, a_2, a_3, …, a_n is a multiset of directed arcs, a_i = (v_j, v_k), with v_j, v_k ∈ V. The set V can be partitioned into two disjoint sets P and T such that V = P∪T, P∩T = Φ, and for each directed arc, a_i ∈A, if a_i = (v_j, v_k), then either v_j ∈ P and v_k∈ T or v_j ∈ T and v_k ∈P.
Furthermore, the formal definition of a Coloured Petri Net is as follows: A non-hierarchical coloured Petri net is a tuple CPN = (Σ, P, T, A, N, C, G, E, I) satisfying the requirements below: (1) Σ is a finite set of non-empty types, called *colour sets*. (2) P is a finite set of *places*. (3) T is a finite set of *transitions*. (4) A is a finite set of arcs such that: $P \bigcap T = P \bigcap A = T \bigcap A = \varnothing$. (5) N is a *node* function. It is defined from A into $P \times T \bigcup T \times P$. (6) C is a *colour* function. It is defined from P into Σ. (7) G is a *guard* function. It is defined from T into expressions such that: $\forall t \in T : [Type(G(t)) = B \wedge Type \quad (Var(G(t))) \subseteq \Sigma]$. (8) E is an *arc expression* function. It is defined from A into expressions such that: $\forall a \in A : [Type(E(a)) = C(p(a))_{MS} \wedge Type(Var(E(a))) \subseteq \Sigma]$ where p(a) is the place of N(a). (9) The I is an *initialization* function. It is defined from P into closed expressions such that: $\forall p \in P : [Type(I(p)) = C(p)_{MS}]$.
The formal definition of timed Coloured Petri Nets (Jensen 1997), i.e., the formal definition of Stochastic Coloured Petri Net, is as follows: A *timed* non-hierarchical Coloured Petri Net is a tuple TCPN = (CPN, R, r_0) such that (1) Coloured Petri Net satisfied the requirements of a non-hierarchical Coloured Petri Net as defined in the abovesection when in arc expression function and the initialization function. We allow the type of E(a) and I(p) to be a timed or an un-timed multiple set over C(p(a)) and C(p), respectively. (2) R is a set of *time values*, also called *time stamps*. It is a subset of ℜ closed under + and containing 0. (3) r_0 is an initial element of R, called the *start time*.

The interval definition is as follows: (1) TS is the time set, $TS = \{x \in \Re \mid x \geq 0\}$, i.e. the set of all non-negative real numbers. (2) $INT = \{[y,z] \in TS \times TS \mid \leq z\}$, represent the set of all closed intervals. If $x \in TS$ and $[y,z] \in INT$ then $x \in [y,z]$ if and only if $y \leq x \leq z$.

The basic elements of a CSPN graph are listed as follows (Haas 2002): (1) A finite set $D = \{ d_1, d_2, \ldots, d_L \}$ of *places*. (2) A finite set $E = \{ e_1, e_2, \ldots, e_M \}$ of *transitions*. (3) A (possibly empty) set $E' \subset E$ of *immediate transitions*. (4) A finite set U of C*olours* with a fixed enumeration. (5) Colour domains UD(d) $\subseteq$ U for d $\in$ D and UE(e) $\subseteq$ U for e $\in$ E. (6) An input incidence function w− and an output incidence function w+, each defined on $U_{e \in E, d \in D}$ ({e}× UE(e)×{d} × UD(d)) and taking values in the nonnegative integers.

2.3 Particle Filter

The particle filter is an inference technique that estimates the unknown state from the sampling particle collection of observation $Y_{1:t}=\{Y_1, \ldots, Y_t\}$. It approximates the posterior distribution $p(S_t \mid Y_{1:t})$ by a set of weighted particles $Z_t = \{Y_t^{(i)}, w_t^{(i)}\}_{i=1}^N$ with $\sum_{i=1}^{N} w_t^{(i)} = 1$.

The dynamic state system consists of the state transition model and the observation model. The s*tate transition model:* $S_t = F_t(S_{t-1}, V_t)$, and the o*bservation model:* $Y_t = H_t(S_t, W_t)$. The state transition function F_t approximates the dynamics of the object being tracked using the previous state S_{t-1} and the system noise V_t, and the measurement function H_t models a relationship between the observation Y_t and the state S_t given the observation noise W_t. We usually characterize the state transition model with the state transition probability $p(S_t \mid S_{t-1})$ and the observation model with the likelihood $p(Y_t \mid S_t)$.

A general procedure of the particle filter consists of three steps: re-sampling, prediction, and update step. In the re-sampling step, we resample the particles Z_{t-1} to obtain the non-weighted set of particles with equal weights $\{S_{t-1}^{'(i)}, 1\}_{i=1}^N$. In the prediction step, we draw the particles $\{V_t^{(i)}\}_{i=1}^N$ and generate the particles $\{S_{t-1}^{'(i)}, 1\}_{i=1}^N$ using the state transition model $S_t = F_t(S_{t-1}, V_t)$. In the updating step, we update the weight of each particle based on the observation likelihood as $w_t^{(i)} \propto p(Y_t \mid S_t^{(i)})$.

Particle filter is entitled by a group of weighting particles to calculate posterior probability. The Equation (1) is a formula of the Bayes theorem of posterior probability.

$$P(S_t \mid Z_{1:t}) = \frac{P(Z_{1:t} \mid S_t)P(S_t)}{P(Z_{1:t})} = \frac{P(Z_t\ , Z_{1:t-1} \mid S_t)P(S_t)}{P(Z_t\ , Z_{1:t-1})}$$

$$= \frac{P(Z_t \mid Z_{1:t-1}\ , S_t)P(Z_{1:t-1} \mid S_t)P(S_t)}{P(Z_t \mid Z_{1:t-1})P(Z_{1:t-1})}$$

$$= \frac{P(Z_t \mid Z_{1:t-1}\ , S_t)P(S_t \mid Z_{1:t-1})P(Z_{1:t-1})P(S_t)}{P(Z_t \mid Z_{1:t-1})P(Z_{1:t-1})P(S_t)} \quad (1)$$

$$= \frac{P(Z_t \mid S_t)P(S_t \mid Z_{1:t-1})}{P(Z_t \mid Z_{1:t-1})}$$

In the Equation (1), we can find that the posterior probability $P(S_t \mid Z_{1:t})$ could be presented by the priori probability $P(S_t \mid Z_{1:t-1})$ and the observation model $P(Z_t \mid S_t)$. Therefore, we can obtain the Equation (2) and (3).

$$\begin{aligned} P(S_t \mid Z_{1:t-1}) &= \int P(S_t, S_{t-1} \mid Z_{1:t-1})dS_{t-1} \\ &= \int P(S_t \mid S_{t-1}\ , Z_{1:t-1})P(S_{t-1} \mid Z_{1:t-1})dS_{t-1} \\ &= \int P(S_t \mid S_{t-1})P(S_{t-1} \mid Z_{1:t-1})dS_{t-1} \end{aligned} \quad (2)$$

$$\begin{aligned} P(Z_t \mid Z_{1:t-1}) &= \int P(Z_t, S_{t-1} \mid Z_{1:t-1})dS_{t-1} \\ &= \int P(Z_t \mid S_t\ , Z_{1:t-1})P(S_t \mid Z_{1:t-1})dS_{t-1} \\ &= \int P(Z_t \mid S_t)P(S_t \mid Z_{1:t-1})dS_{t-1} \end{aligned} \quad (3)$$

Therefore, the Equation (2) and (3) substitute for (1) , we can obtain the Equation (4).

$$P(S_t \mid Z_{1:t}) = kP(Z_t \mid S_t)\int P(S_t \mid S_{t-1})P(S_{t-1} \mid Z_{1:t-1})dS_{t-1}\text{ , where } k \text{ is a normalized constant.} \quad (4)$$

According to the Equation (4), we can evaluate and forecast the real state of the object. $P(Z_t \mid S_t)$ is the observation model of the probability density function that is a likelihood function. $P(S_t \mid S_{t-1})$ is the state transform model and $P(S_{t-1} \mid Z_{1:t-1})$ is the posterior probability at time t-1. So that, we can substitute and update each state of posterior probability of object along with the initial state distribution $P(S_0)$. As $N \to \infty$, the Equation (2) can be presented by the Equation (5).

$$\begin{aligned} P(S_t \mid Z_{1:t-1}) &= \int P(S_t \mid S_{t-1})P(S_{t-1} \mid Z_{1:t-1})dS_{t-1} \\ &\approx \sum_{i=1}^{N} P(S_t \mid s_{t-1}^{(i)})w_{t-1}^{(i)} \end{aligned} \quad (5)$$

Therefore, the Equation (2-8) could be substituted by the Equation (6).

$$P(S_t \mid Z_{1:t}) = kP(Z_t \mid S_t)\int P(S_t \mid S_{t-1})P(S_{t-1} \mid Z_{1:t-1})dS_{t-1}$$
$$\approx kP(Z_t \mid S_t)\sum_{i=1}^{N} P(S_t \mid s_{t-1}^{(i)})w_{t-1}^{(i)} \quad (6)$$

The principal steps in the particle filter algorithm:
// Input: the object that would be analyzed, detected and traced.

// Output: a set of particles $\{S_t^{(i)}, w_t^{(i)}\}_{i=1}^{N}$ that can be used to approximate the posterior distribution.
Step 1: Initializing particles.
Set t = 1;

Generate particle set from the initial distribution $p(S_0)$ to obtain $\{S_0^{(i)}, w_0^{(i)}\}_{i=1}^{N}$, the initial state N particles $\left\{s_0^{(i)}\right\}_{i=1}^{N}$ and setting their weights $\left\{w_0^{(i)}\right\}_{i=1}^{N}$, where each $w_0^{(i)} = 1/N$.
Step 2: The forecasting the next state of particles.

In the set of particles, $\left\{s_t^{(i)}\right\}_{i=1}^{N}$ presents the state of each particle at time t, according to the transition model $p(S_k^{(i)} \mid S_{k-1}^{(i)})$.
Step 3: Observing and evaluating particles.
Evaluate the new state of detected particles by the importance likelihood:

$w_t^{(i)} = \dfrac{w_t^{(i)}}{\sum_{j=1}^{N} w_t^{(j)}}$ $i = 1,...,N$. Let the new weights at time t be $w_t^{(i)} = P(Z_t \mid S_t^{(i)})$ $i = 1...N$.

Step 4: Output

Output a set of particles $\{S_k^{(i)}, w_k^{(i)}\}_{i=1}^{N}$ that can be used to approximate the posterior distribution as $p(S_t \mid Z_t) = \sum_{i=1}^{N} w_t^{(i)}\delta(S_t - S_t^{(i)})$ where δ () is the Dirac delta function.

Step 5: Resample

Resample particle set $\left\{s_t^{(i)}\right\}_{i=1}^{N}$ with probability $w_t^{(i)}$ to obtain *N* independent and identically distributed random particle set $\left\{s_t^{(j)}\right\}_{j=1}^{N}$ approximately distributed according to $p(S_t \mid Z_t)$. The resample sub-algorithm is as follows.

Let $New\_w_t^{(1)} = w_t^{(1)}$;
For i = 2 : N
$New\_w_t^{(i)} = New\_w_t^{(i-1)} + w_t^{(i)}$
End for

For i = 1 : N
  r = random(0,1);
  for j = 1 : N
    if ($New\_w_t^{(j)} \geq r$) then
      k = j;
      break for;
    end if
  end for
  if (i <> k) then
    $New\_s_t^{(i)} = s_t^{(k)}$
  End if
End for

Step 6 Set t = t + 1, and return to Step 2.

3. The NetworkParticle Filtering Model in Intrusion Detection

A time window is during three seconds. Moving a time window per one second or two seconds, i.e., there are two one or two seconds overlap between two time windows. We Used thenetwork particle filter scheme to classify network packets into two classes those include normal or abnormal network behaviors in each time window. To classify packets within the continue time window is to classify packets in each time window during a longer time. The system builds the relationship within these classes. The basic information in each network packet includes source IP, destination IP, source TCP port number and destination TCP port number.

3.1 The Definition of the Network Intrusion

Firstly, we give some definitions to describe the meanings and behaviors of network flow in IDSs.

Definition 1 Time Window: A time window is a time interval that covers many network flow packets.

Definition 2 Malicious Event: An event generated by a single attempt to violate certain security policies, regardless of whether the attempt achieves its goal.

According to definition 2, even if an attempt fails to violate a security policy, the events it generates are still malicious. This conforms to the common understanding of a malicious event. For example, an attempt to overflow a buffer on no vulnerable web server is still malicious, even though it fails.

Definition 3 Suspicious Event: No malicious event generated by an attempt that has a strong logical connections with the malicious events.

For example, some Snort signatures detect IP sweep attempts that do not violate the security policies of many sites. However, these events often have a strong connection to intrusion attempts because the attackers are trying to identify active computer systems.

Definition 4 Attack: A malicious or suspicious event detected by the IDSs.

We shall concentrate on the events that IDSs detect, because usually attacks are only discernable in terms of IDS alerts. Moreover, alert correlation only works on the alerts, and not

on the events that the IDSs do not detect. In addition, this definition of an attack makes it interchangeable with the IDS alert in the following. Thus, we will not always explicitly state that an attack is represented by the alerts.

Definition 5 Alert: A message reported by the IDSs as the result of an attack.

Definition 6 Intrusion Incident: A sequence of related attacks within a time frame against a single computer system by an attacker to achieve some goals.

The definition 5 and 6 describe the output of the IDS. The Alert can talk to a system or system manager to make a response for this Alert automatically or artificially.

Definition 7 Alert Fusion (Aggregation): Grouping alerts by their common characteristics; typically, grouping alerts of the same signature and network addresses.

Definition 8 Requires/Provides (Prerequisite) Relation: If an early attack provides logical support, e.g., information of or access to the system under attack, for a later attack that requires it, there is a requires/provides relation between the two attacks and the corresponding alerts of the attacks.

Definition 9 Alert Correlation: Grouping alerts by their required or provided relation.

The definitions 7, 8 and 9 are to analyze and build the relationship between Alerts, and then the IDS can supply more useful report or response policies.

3.2 Network Particle Filtering Model

The Fig. 1 shows the proposed network particle filtering model that is implemented in the CPN tools. In this model includes senders who send network packetsto some hosts. The NPF recognizes and classifies each network packet into normal or abnormal classes by network particle filter scheme. Those hosts are the destination computer of the sent network packet from a sender.

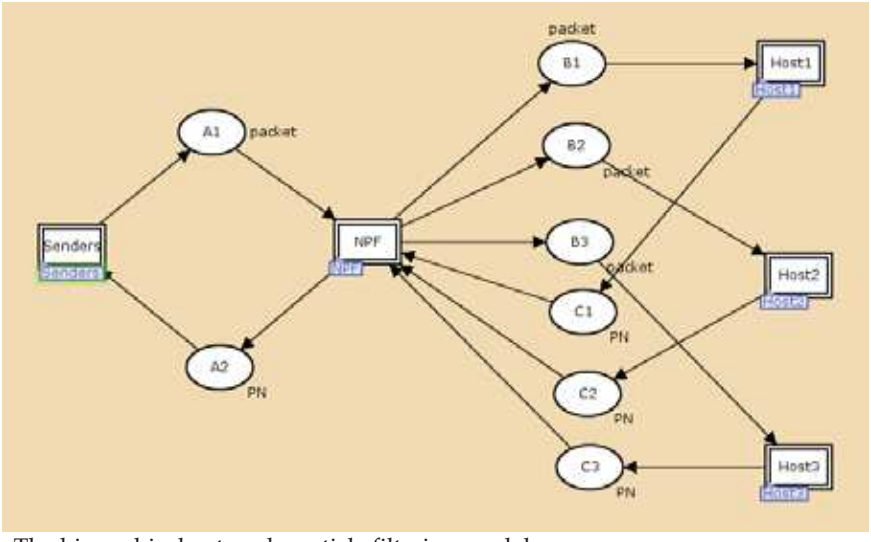

Fig. 1. The hierarchical network particle filtering model.

In this research, we considered four cases to analyze network packets and two kinds of attacksto detect intrusion behaviors by network particle filtering model. The first case is one

packet analysis, to consider each packet, and select N features as particles. Each particle is given a different weight. Through multi-step filtering, each packet could be defined as normal or abnormal behavior. If one packet is found to be an abnormal behavior, then tracking corresponding packages with the same as source IP address, TCP port number, and UDP port number and increasing the corresponding weight of particles. The second case is multiple packet analysis, the timed packets flow. To find and analyze the relationship between multiple packets, how to decide each next related packet is normal or abnormal behavior. To increase normal particle weight and decrease abnormal particle weight when the last related packet is belonged to normal behavior. On the other hand, to decrease normal particle weight and increase abnormal particle weight when the last related packet is belonged to abnormal behavior. Then evaluate the score for each related packet particle. If the normal score is greater than the abnormal score, then this packet is belonging to the normal behavior. In the similar, if the abnormal score is greater than the normal score, then this packet is belonging to the abnormal behavior.

The next case is in a time window, the number of the source and destination IP address (NSDIP) and the number of source and destination TCP port (NSDTCP) for each packet would be summarized and given a weight and probability for each NSDIP and NSDTCP in a time window. The value of weights is between 0 and 1, and their sum is equal to 1. A threshold of the weight would be given to evaluate whether some packets are abnormal behaviors or not.

The final case is within multiple time windows and overlapping time windows. The next step, we selected those abnormal packets from multiple time windows. Those abnormal packets would be analyzed and found the relationships betweenthem. Therefore, those abnormal packets would be classified and named one attack. And then IDS creates the attack pattern and update into the pattern database. The IDSs could make a response to each detected attack. For example, IDSs could send alerts to system manager, log each detected incident and attack, and auto-response by system defined.

We assume that there are some relationships for some packets between two neighbor time windows. If the relationship exists, then we can work at the packet trace. Otherwise, it will be failed to trace. Therefore, we should extend the filtering field to more time windows that maybe cover some packets with relationship; or begin another packet trace because the last trace packet is the end of sequence.

The detection models can be divided into offline and online cases. Fig. 2 (a) shows the flowchart of offline detection case. In the offline case, the input is the collected data during a time interval that included more than one time window. The next step is to analyze and classify packets during one time window using the network particle filter scheme. And then the step is to detect intruded behaviors and update the intrusion pattern database when the new intrusion behaviors were been found.

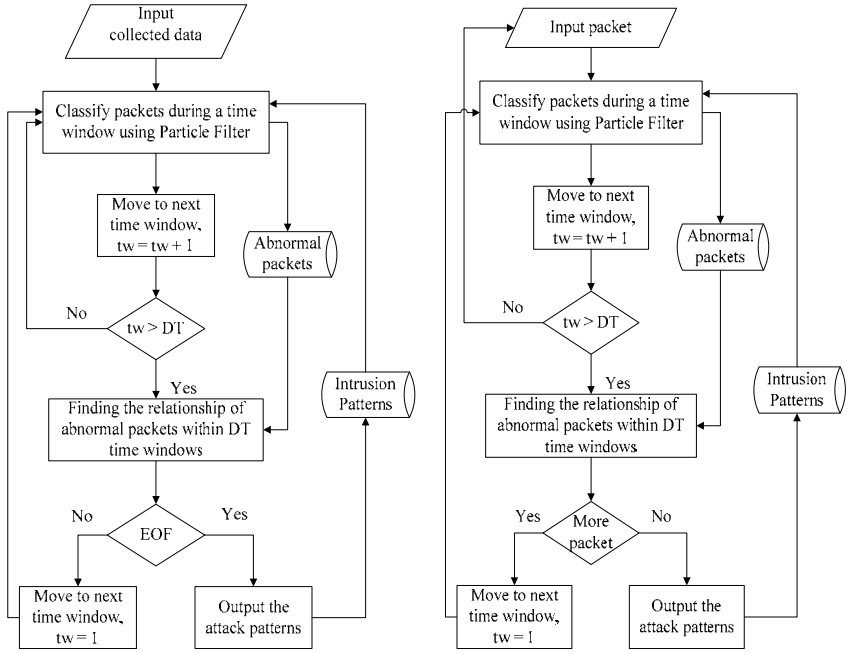

Fig. 2. (a) The offline analysis flowchart of the network particle filtering IDS. (b) The online detection flowchart of the network particle filtering IDS.

Fig. 2 (b) shows the flowchart of offline detection case. The most difference between offline and online cases is the input data. In the online case, the input data are the real time received network packets from senders. Therefore, we analyze and classify packets using the network particle filter scheme when the time is up for a time window. The abnormal packet database is to keep the dubious packets those could be used after some time windows. The intrusion pattern database is to save the patterns that had been confirmed as intrusionbehaviors. The intrusion pattern database could be updated when the system found a new intrusion pattern.

4. Experimental Results

In our experiment include two simulation cases Intrusion detection and Trojan detection. In the Intrusion detection case, we assume that the almost intrusion behaviors come from senders. Therefore, we just design the network particle filter scheme to detect the packets those have been sent from senders. On the other hand, the Trojan detection case, we assume almost dubious packets come from receiver's acknowledge. Therefore, we set the network particle filter scheme on the outward path. The network particle filter scheme is designed to analyze and classify each network packet into normal or abnormal class for inward and outward, respectively.

The simulation platform is CPN Tools for Coloured Petri Nets that supports good interface and tools to implement the Coloured Petri Nets model. We design a hierarchical network that includes four main parts system view, sender, NPF, and hosts. We also let each kind attack simulation be executed 4000 steps to claim the trend of the results.

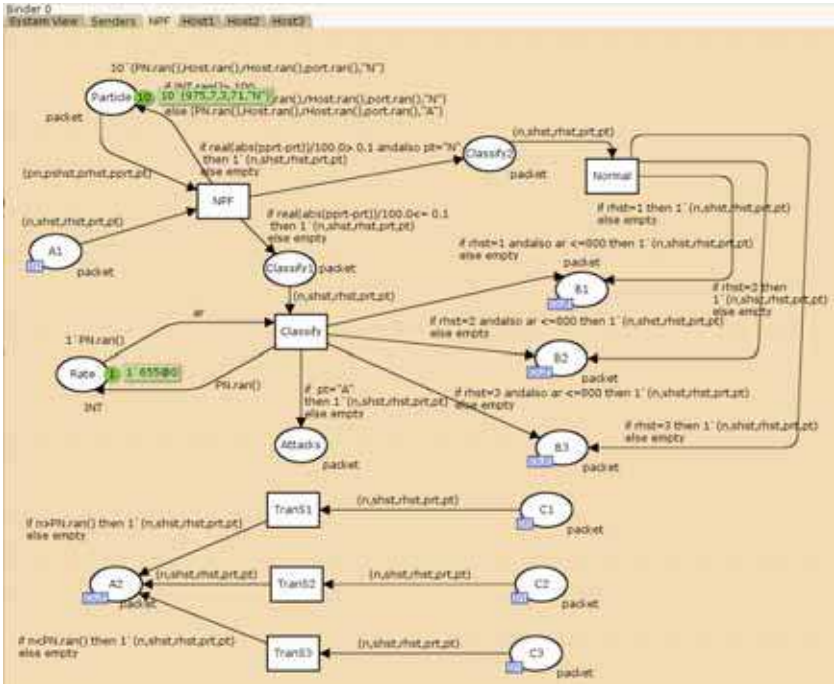

Fig. 3. The CPN simulation for NPF-Intrusion. The initial status.

Fig. 3 shows the initial status of NPF part in the Intrusion detection case using CPN Tools. The 'Particle' place is to be designed to create 10 particles randomly. The 'NPF' transition is to set up the filtering conditions and classify packets into normal class or dubious class. The 'Classify' transition refines the classification of 'NPF'. And then sends the attack packets to the 'Attacks' place. The 'Attacks' place is to record the attack packets those have been captured by 'NPF' transition.

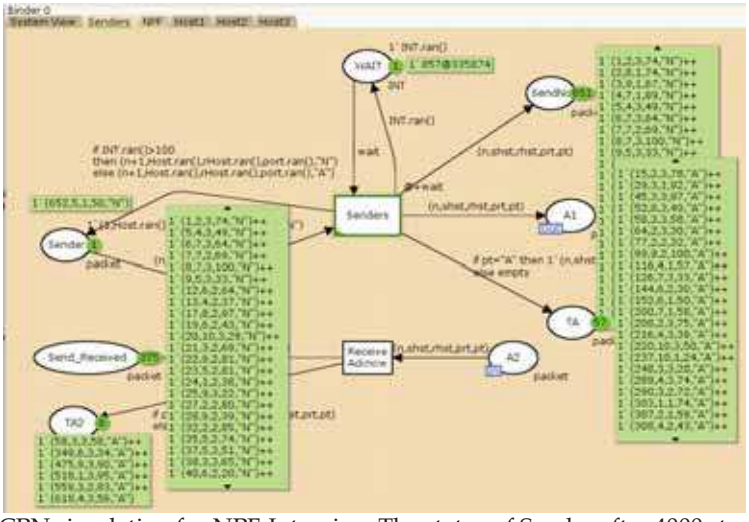

Fig. 4. The CPN simulation for NPF-Intrusion. The status of Sender after 4000 steps.

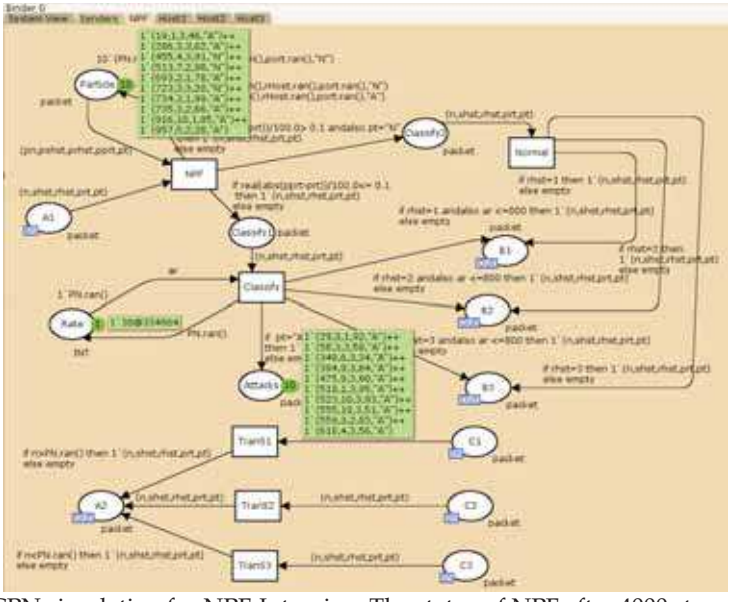

Fig. 5. The CPN simulation for NPF-Intrusion. The status of NPF after 4000 steps.

Fig. 5 shows the status of NPF part after 4000 steps. There are 10 attack packets in 'Attacks' place, i.e., the system captured 10 attack or intrusion behaviors. At the same time, there are 57 attack packets in 'TA' place. So that, we can obtain the total detection rate is 17.54%.
Fig. 6 shows the initial status of NPF part for the Trojan detection case. The 'Collect' place is to collect all acknowledged packets from the receiver , and then sends them to the 'NPF' transition. The 'NPF' transition is to detect each passed packet is the Trojan behavior or not.

The 'Trojan' place saved the possible Trojan packets those have been captured from the 'NPF' transition.

Fig. 7 shows the status of the Sender part after 4000 steps on the Trojan detection case. The total sent packets are 693. At the same time, there are 470 acknowledged packets have been received. Fig. 8 shows the status of NPF part after 4000 steps on the Trojan detection case. The total number of Trojan packets is 66 in the 'Trojan' place. So that, we can obtain the total Trojan behavior rate is 14.04%.

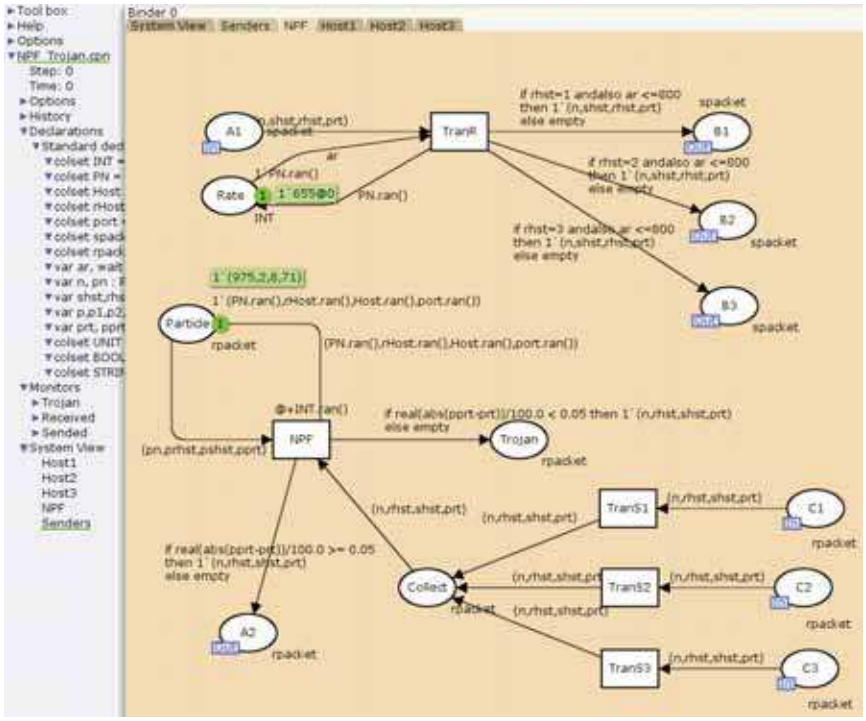

Fig. 6. The CPN simulation for NPF-Trojan. The initial status.

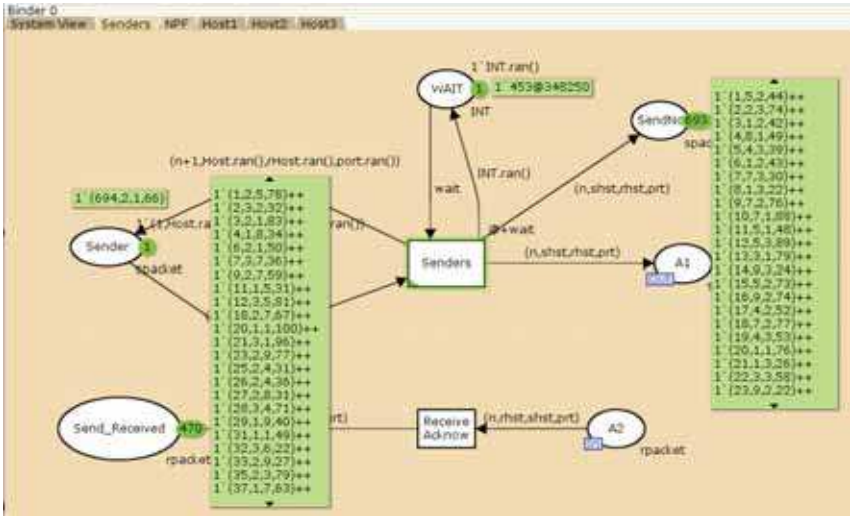

Fig. 7. The CPN simulation for NPF-Trojan. The status of Sender after 4000 steps.

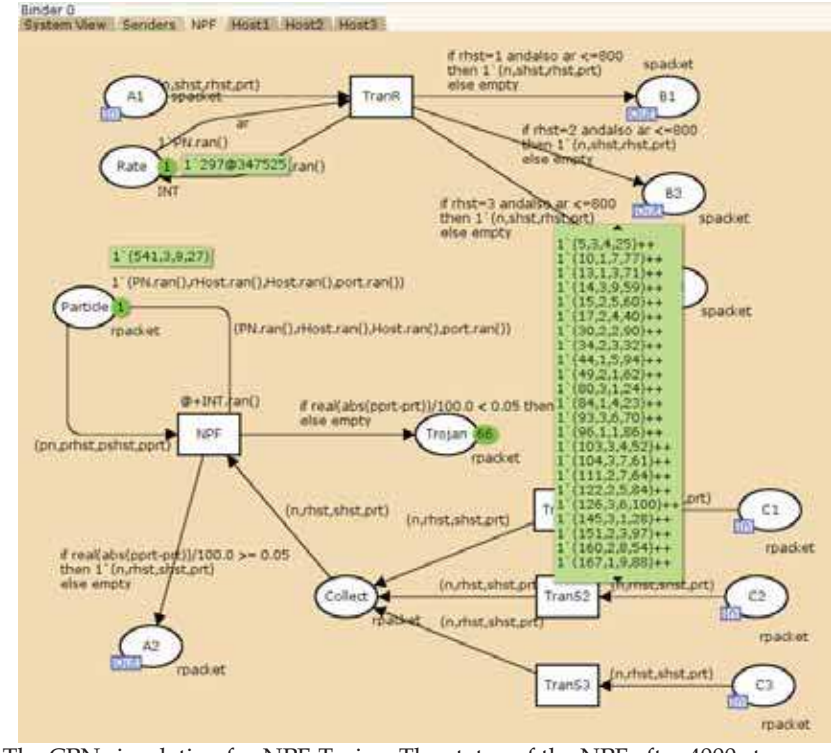

Fig. 8. The CPN simulation for NPF-Trojan. The status of the NPF after 4000 steps.

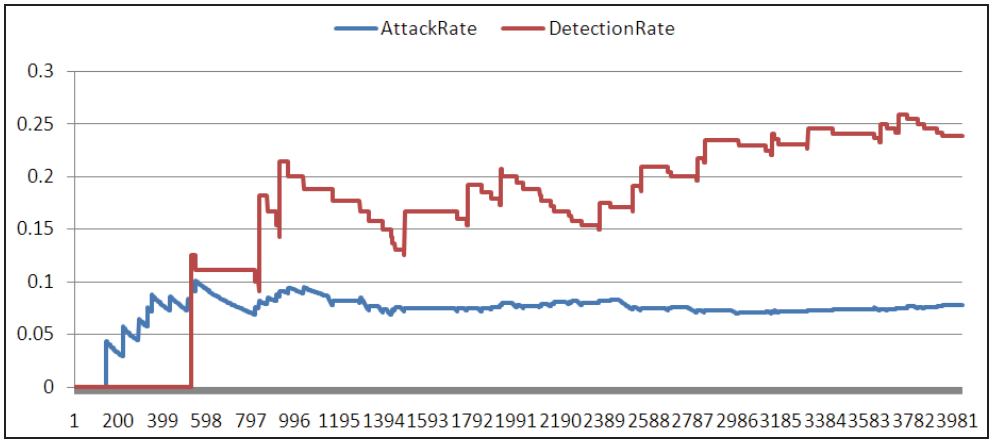

Fig. 9. The attack rate and detection rate of the CPN simulation for NPF-Intrusion after 4000 steps.

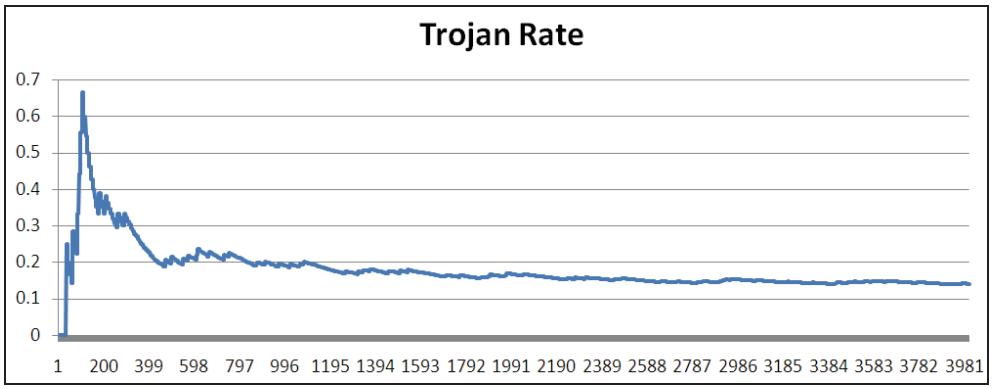

Fig. 10. The Trojan rate of the CPN simulation for NPF-Trojan after 4000 steps.

In our simulation process, we record the number of some places, for example, "sendNo", "Attacks" and „Send_Received" at eachstep. Fig. 9 shows the attack packet rate and attack detection rate. The attack packet rate is the ratio of attack packets against all sent packets. The attack detection rate is the ratio of attack packets against all signed 'A' packets. Fig. 10 shows the Trojan rate that is the ratio of Trojan packet against all acknowledged packets from the receivers. In the first 200 steps illustrates a suddenly rise high interval, the reason is the most of the acknowledged packets passed through the NPF node but have not yet arrived at the sender node. The trend, however, of the results matches our design.

5. Conclusions

In this paper, we have provided a network particle filtering with a stochastic model for an intrusion detection system, and simulated this scheme in the ColouredPetri Nets Tools plat-

form. To build a real test environment and collect real attack case data is not easy. We proposed a test bed platform that can support to test and simulate the network attack cases.
The particle filter scheme applied into network analysis is a difficult work, because the related behaviors of network flow are not continuous and it is very difficult to know and control them. Therefore, the accuracy of our simulation results is not enough good. The design of network flow also does not touch the real attack cases. Our approaches, however, can be applied to the risk and cost evaluation to practice an IDS.

6. References

Dahl, O. M. (2005). Using Coloured Petri Nets in Penetration Testing. Department of Computer Science and Media Technology. Gjøvik, Gjøvik University College. Master: 1-89.

Dahl, O. M. and S. D. Wolthusen (2006). Modeling and execution of complex attack scenarios using interval timed colored Petri nets. the Fourth IEEE International Workshop on Information Assurance.

Gordon, N. J., D. J. Salmond, et al. (1993). "Novel approach to nonlinear/non-Gaussian Bayesian state estimation." Inst. Elect. Eng. F, Radar Signal Process 140: 107-113.

Haas, P. J. (2002). Stochastic Petri Nets Modelling, Stability, Simulation Springer-Verlag New York, Inc.

Hurzeler, M. and H. R. Kunsch (1998). "Monte Carlo approximations for general state space models." J. Computat. Graph. Statist 7(2): 175-193.

Isard, M. and A. Blake (1998). "CONDENSATION—Conditional Density Propagation for Visual Tracking." Int. J. Comput. Vision 29: 5-28.

Jensen, K. (1992). Coloured Petri Nets Basic Concepts, Analysis Methods and Practical Use Springer-Verlag.

Jensen, K. (1997). Coloured Petri Nets Basic Concepts, Analysis Methods and Practical Use Springer-Verlag.

Kotecha, J. H. and P. M. Djuric (2003). "Gaussian sum particle filtering." IEEE Trans. Signal Process 51(10): 2602-2612.

Kristensen, L. M. and K. Jensen, Eds. (2004). Specification and Validation of an Edge Router Discovery Protocol for Mobile Ad-hoc Networks. Lecture Notes in Computer Science, Springer-Verlag.

Kristensen, L. M., J. B. Jorgensen, et al., Eds. (2004). Application of Coloured Petri Nets in System Development. Lecture Notes in Computer Science, Springer-Verlag.

Lehn-Schioler, T., D. Erdogmus, et al. (2004). "Parzen particle filters." IEEE Int. Conf. Acoust., Speech, Signal Process 5: 781-784.

Pitt, M. and N. Shephard (1999). "Filtering via simulation: Auxiliary particle filters." J. Amer. Statist. Assoc. 94(446): 590-599.

Savage, S., D. Wetherall, et al. (2001). "Network Support for IP Traceback." IEEE/ACM TRANSACTIONS ON NETWORKING 9(3): 226-237.

Snoeren, A. C., C. Partridge, et al. (2002). "Single-Packet IP Traceback." IEEE/ACM TRANSACTIONS ON NETWORKING 10(6): 721-734.

Steffan, J. and M. Schumacher (2002). Collaborative attack modeling. Proceedings of the 2002 ACM symposium on Applied computing, Madrid, Spain, ACM Press.

Petri Nets Applications

Edited by Pawel Pawlewski

ISBN 978-953-307-047-6
Hard cover, 752 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

Petri Nets are graphical and mathematical tool used in many different science domains. Their characteristic features are the intuitive graphical modeling language and advanced formal analysis method. The concurrence of performed actions is the natural phenomenon due to which Petri Nets are perceived as mathematical tool for modeling concurrent systems. The nets whose model was extended with the time model can be applied in modeling real-time systems. Petri Nets were introduced in the doctoral dissertation by K.A. Petri, titled "„Kommunikation mit Automaten" and published in 1962 by University of Bonn. During more than 40 years of development of this theory, many different classes were formed and the scope of applications was extended. Depending on particular needs, the net definition was changed and adjusted to the considered problem. The unusual "flexibility" of this theory makes it possible to introduce all these modifications. Owing to varied currently known net classes, it is relatively easy to find a proper class for the specific application. The present monograph shows the whole spectrum of Petri Nets applications, from classic applications (to which the theory is specially dedicated) like computer science and control systems, through fault diagnosis, manufacturing, power systems, traffic systems, transport and down to Web applications. At the same time, the publication describes the diversity of investigations performed with use of Petri Nets in science centers all over the world.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821