Pace University
DigitalCommons@Pace

CSIS Technical Reports

Ivan G. Seidenberg School of Computer Science and Information Systems

10-16-2005

Intrusion Detection and Response System Generator

Bel G. Raggad
Ivan G.Seidenberg School of Computer Science and Information Systems

Follow this and additional works at: http://digitalcommons.pace.edu/csis_tech_reports

Recommended Citation
Raggad, Bel G., "Intrusion Detection and Response System Generator" (2005). *CSIS Technical Reports.* Paper 19.
http://digitalcommons.pace.edu/csis_tech_reports/19

This Article is brought to you for free and open access by the Ivan G. Seidenberg School of Computer Science and Information Systems at DigitalCommons@Pace. It has been accepted for inclusion in CSIS Technical Reports by an authorized administrator of DigitalCommons@Pace. For more information, please contact rracelis@pace.edu.

SCHOOL OF COMPUTER SCIENCE AND INFORMATION SYSTEMS

TECHNICAL REPORT

Number 213, April 2005

Intrusion Detection and Response System Generator

Bel G. Raggad

PACE UNIVERSITY

Bel G. Raggad is Professor of Information Systems at Pace University, based in Westchester. He holds the Ph.D. from Pennsylvania State University and has been a full-time member of the Pace faculty since 1996.

Professor Raggad spent the 2003-2004 academic year as a Fulbright scholar in Tunisia. Before that, he was among eight international experts selected by the United Nations Industrial Development Organization (UNIDO) to advise on industrial restructuring and Internet security in developing countries.

Much of Professor Raggad's research and many of his over one hundred articles and books are in the area of information assurance.

Intrusion Detection and Response System Generator

Bel G. Raggad, Ph.D.
IS Department, CSIS
Pace University
Pleasantville, NY 10570
braggad@pace.edu

Abstract

We discuss the design of an intrusion detection and response system (idrs) generator. This design involves a belief fuser, a belief tree classifier, and a memoryless fuzzy incident responder. The firm's security policy, its current risk profile, and training data constitute input streams. The system is designed to produce an incident response that security officers feasibly adopt to improve the firm's risk position as indicated in the corporate security policy. We do not present a prototype for the idrs generator but we provide sufficient details on the credal and pignistic schemes for the fuser and the classifier, needed to develop the idrs generator.

Keywords: fuser, classifier, transferred belief model, credal model, pignistic probability, belief, fuzzy rule base.

Introduction

Intrusion detection is important capability in information assurance. Any security system remains partial without the support of this capability. Massive data processing, in intrusion detection, slows the identification of the intrusion and delays the planning of any incident response action.

Intrusion detection often involves combining multiple sources of information which is, despite the profusion of statistical research, still a major and difficult task in the management of uncertainty. But full security is really impossible to maintain in a computing environment. Security officers, who may know all possible threats, all possible vulnerabilities, and all available security controls, still cannot make an accurate projection of all these factors on their computing environment, without thorough and costly testing activities. Security officers can only develop belief models about the type of intrusions threatening the system. It is impossible therefore to develop the dual belief model on the non-occurrence of any type of intrusion, which expresses the amount of ignorance involved in the security officer's evidence structure.

Under these conditions, Dempster and Shafer's theory should apply. We will however assume that sensors work independently, which is a very reasonable assumption that can be easily achieved by configuring the ids reporting system in this manner. This way we can then prevent the computing complexity imposed by incidence calculus needed to combine evidence generated by dependent sources.

Table 1: Design tasks studied	
Design tasks	Descriptions
Sensors configurations	The following 8 configurations are examined 1) fully disjunctive; 2) fully conjunctive; 3) partially disjunctive; 4) partially conjunctive; 5) bijunctive; 6) random disjunctive; 7) random conjunctive; and 8) random bijunctive.
Fuser	n sensors generate n basic belief assignments. The fuser combines evidence according to the 8 configurations listed above. The fuser also discounts evidence based on sensors reliability expressed as a basic belief assignment.
Classifier	An induced belief tree will be used as a classifier. This tree is grown based on the maximization of Shannon's information gain expressed in terms of pignistic probabilities obtained from belief functions. A training data set will be used to grow the tree and a testing data set will be used in post-pruning for the elimination of over fitting.
Responder	A rule base subsystem will process the output of the classifier and produce the security controls that reduce risks below a tolerated level according to corporate security policy.
Demonstration	A prototype will not be developed, but numerical examples will be worked out to the designer satisfaction in terms of the mini-specifications needed to develop the idrs generator.

This article discusses the design of an intrusion detection system equipped with a incident response system (idrs). This study will limit design features to those components listed in Table 1. An experimental framework is given in Figure 1.

Before we further proceed, let us introduce some notations. Let Ω be our frame of discernment for our sensors' outputs. Also let B be a Boolean algebra of subsets of Ω. The degree of belief held by a sensor S at time t that the actual state $\omega 0$ belongs to the set A of states is equal to x, where A is a subset of the frame of discernment Ω and $A \in B$ is:

$$Bel_{\{\Omega.B.S.t\}}[e(S,t)]\ \ (w0 \in A) = x.$$

The belief is based on the evidential corpus e(S,t) held by S at t, where e(S,t) represents all what the sensor S knows at t. Even though this notation is general and allows for a dynamic system, this study will be limited to one instantiation of the sensor reporting system. The idrs generator is hence memoryless, for it does not allow for combining past data with the current sensors' reports. This is not in any way meant to be a statefull inspection system because we do not include the extraction and propagation processes, and limit ourselves to the combination of evidence alone.

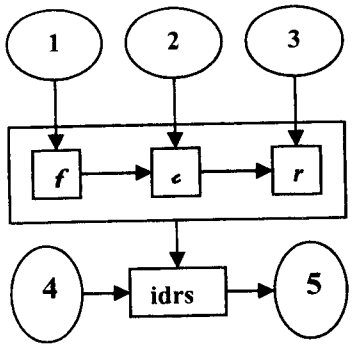
We will soon omit some of the subscripts to ease our notation style. Most often, B is actually the Boolean algebra 2^{Ω}, the power set of Ω. When B is not explicitly stated, it

means that Bel is defined on 2^{Ω}. Also 'ω0 ∈ A' is often denoted as simply 'A'. When the missing elements are clearly defined from the context, S, t, Ω and other parameters will be left out as needed. So $Bel_{\{\Omega\}}[E](A)$ will sometimes be simply denoted as Bel(A).

That is, $Bel_{\{\Omega.B.S.t\}}[e(S,t)]$ which denotes the belief function should be viewed as as a finite vector of length |B|, with its components formed by the values of $Bel_{\{\Omega.B.S.t\}}[e(S,t)]$ (A) for every A in Ω. Using our notation, it should be clear that it also applies to the bba, plausibility, and other functions, as in $m_{\{\Omega.B.S.t\}}[e(S,t)]$ (w0 ∈ A), $pl_{\{\Omega.B.S.t\}}[e(S,t)]$ (w0 ∈ A), respectively.

Let g denote our target idrs generator. The idrs generator's design composes three main components: a fuser f, a classifier c, and an incident response module r. The idrs is hence equipped with a fuser f which receives all sensors' messages and processes them to produce a fused message. One may consequently write $g \equiv idrs(f(\mathcal{D}), c(\mathcal{D}), r(\mathcal{C}))$, where the symbol '≡' expresses the generator's design delivery requirements to produce an incident response given a fused message.

$g: \mathcal{D} \rightarrow \mathcal{R}$,
$f: \mathcal{D}^* \rightarrow \mathcal{D}$
$c: \mathcal{D} \rightarrow \mathcal{C}$
$r: \mathcal{C} \rightarrow \mathcal{R}$
where
$\mathcal{D}$: fuser's input consisting of data patters generated by sensors
$\mathcal{R}$: incident responder's output consisting of security controls
$\mathcal{C}$: classifier's output consisting of intrusion classes.



Peripherals:
1: Sensors configurations
2: Training data
3: Security policy
4: Sensors' reports
5: Incident responses
Subsystems:
f: Fuser
c: Classifier
r: Responder

Figure 1: idrs design

Most specifications of the idrs generator, are defined in the corporate security policy, for intrusion patterns and security controls. All specifications for additional technical requirements should be approved by the security officer before they are added to the design of idrs generator.

The fuser

The fuser accepts sensors' messages (no extraction or propagation processes are implied, as mentioned earlier), combines them, and produces a fused message that the classifier processes to predict the intrusion type for which the responder produces a set of security controls. General design specifications may be discussed in terms of sensors configurations, the fusion process, and the output sent to the classifier. Constraints imposed by sensors configurations and constraints imposed by the classifier's input requirements should be taken into considerations. A fuzzy classifier, for example, requires that the fuser's output be expressed in terms of fuzzy subsets. A possibilistic classifier requires that the fuser's output expressed in terms of possibilities. Traveling from one computing method to another is a central element of the fuser's design specifications.

This article will however adopt a belief tree classifier. The fuser's output stream should, in this case, be written using a belief structure expressed by its basic belief assignments. That is, the total belief fully committed to a subset E in 2^{Ω}, where Ω is the sensors' frame of discernment, is expressed using bel(E) and pl(E) defining the credibility and the plausibility of E, respectively.

$$\begin{aligned} &m: 2^{\Omega} \rightarrow [0,1] \\ &Bel(E)=\sum_{F \leq E} m(F). \\ &Pl(E)= \sum_{F \wedge E \neq \emptyset} m(F). \end{aligned} \qquad (1)$$

This section should discuss the Smets' Transferred Belief Model (TBM) [32, 33] design specifications and computations needed to generate the fuser. Remember, we made the assumption that all sensors are configured to produce Shafer's signals expressed in terms of bba's. Without this assumption, extra computation steps and approximations may be needed to bring the data patterns to a belief structure.

In order to ease interpretability in the fuser's belief structure, we adopt the TBM in two steps: the credal model and the pinistic model [32, 33]. The reader may alternatively opt for Shafer's plausibility functions as a substitute to Smets' pignistic probabilities, as both techniques stem from the same belief structure and both add greater interpretability to the TBM.

The fuser structure at the credal level should look as in Figure 2. A fully asserted evidence model will have the same design without the evidence discount factors $\delta=(\delta_1, \ldots, \delta_n)$.

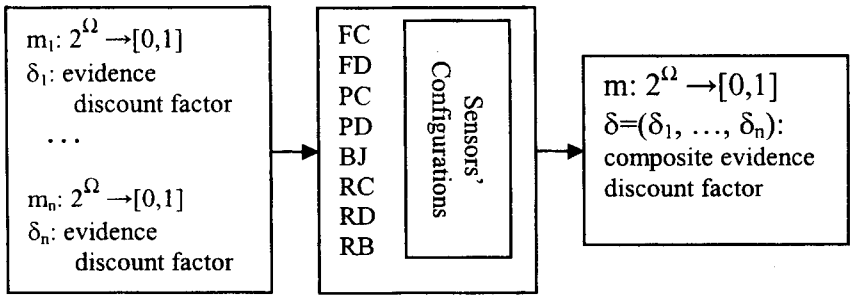


Figure 2: Fuser's design at the credal level with evidence discount

The fuser combines sensors' Shafer's signals and produces the fused Shafer's signal as a one fused bba. In order to grant better interpretability we suggest the credal made of the fused belief structure be transformed into a pignistic model. Alternatively, the security officer can request Shafer's plausibility functions. The plausibility function is computed as Shafer's belief of the subset minus Shafer's belief of its complementary.

At this point, Dempster's rule for combining evidence should apply, at least for both the conjunctive and disjunctive cases, corresponding to sensors' FC and FD configurations. Unfortunately, idrs' sensors configurations do not always allow for the conjunctive and disjunctive cases. Multiple configurations should be considered, for which, as is, Dempster rule does not apply. While we present below most useful sensors configurations, fusing evidence for these cases is beyond the scope of this study and is hence left for another occasion.

The corporate security policy should describe how the idrs components are configured. These configurations may be set, as defined in Table 2, to 1) fully disjunctive (FD); 2) fully conjunctive (FC); 3) partially disjunctive (PD); 4) partially conjunctive (PC); 5) bijunctive (BJ); 6) random disjunctive (RD); 7) random conjunctive (RC); and 8) random bijunctive (RB). These configurations are defined as follows:

Table 2: Sensors' configurations		
Configurations		**Meaning**
FD		Fully disjunctive: Any sensor of $\{s_1, \ldots, s_n\}$ fires.
FC		Fully conjunctive: All sensors in $\{s_1, \ldots, s_n\}$ jointly fire.
PD		Partially disjunctive: Any sensor of $\{s_1, \ldots, s_m\} < \{s_1, \ldots, s_n\}$, where m<n, fires.
PC		Partially conjunctive: All sensors in $\{s_1, \ldots, s_m\} < \{s_1, \ldots, s_n\}$, where m<n, jointly fire.
BJ		Bijunctive: Any sensor of $\{s_1, \ldots, s_m\} < \{s_1, \ldots, s_n\}$, where m<n, fires and all sensors in $\{s_{m+1}, \ldots, s_n\}$ jointly fire.
RD		Random disjunctive: either of m sensors at random fires.
RC		Random conjunctive: all m sensors at random jointly fire.
RB		Random bijunctive: either of m sensors at random fires and all n-m remaining sensors jointly fire.

The credal model:

The design of the creedal step of the idrs generator may be set to fully asserted evidence or discounted evidence.

The case of fully asserted evidence does not discount the evidence induced from a sensor's message. This means that the basic belief assignment expressing the uncertainty associated with the sensor's evidence remain fully asserted. That is:

For any E in Ω, the sensor's frame of discernment, we have:

$$m: 2^{\Omega} \rightarrow [0,1]$$
$$m(\Omega)=1; \sum_{E \leq \Omega} m(E)=1. \qquad (2)$$

Since this sensor's evidence is fully asserted, then Shafer's discount factor equals zero, and the sensor's reliability may be expressed using a belief structure as follows:

$$m(\text{sensor reliability})=1 \qquad (3)$$
$$m(\text{sensor non-reliability})=0.$$

If |s|=1, then there is only one sensor to configure, and the fuser task is reduced to the simple reporting of one sensor. The belief structure is the same as above (1).

The case of discounted evidence imposes a Shafer's discount factor of $1-\delta$ where δ expresses the sensor's reliability. The reliability belief structure is as follows:

$$m(\text{sensor reliability})= \delta \qquad (4)$$
$$m(\text{sensor non-reliability})=1-\delta.$$

If |s|=1, then there is only one sensor to configure, and the fuser task is reduced to the simple reporting of one sensor with discounted evidence. The belief structure is defined as follows:

$$
\begin{aligned}
&m: 2^{\Omega} \rightarrow [0,1] \\
&m(\emptyset)=1;\ \textstyle\sum_{E \leq \Omega} m(E)=1 \\
&\text{For any E in } \Omega, \\
&m^{\delta}(E) = \delta m(E), \text{ and} \qquad (5) \\
&m^{\delta}(\Omega) = (1-.\delta) m(\Omega)
\end{aligned}
$$

In the case where |s|>1, then we have to select the configuration indicated in the corporate security policy from {FD, FC, PD, PC, BJ, RD, RC, RB}. The incorporation of these configurations in the idrs generator's design is beyond the scope of this short paper. We will however demonstrate this design feature using the following approximation, and the FD and DC configuration features:

$|s| = n$

δ= average of reliability factors of sensors with evidence discounted after fusion.

That is, the evidence discount only takes place after the fusion task. This is to say that we first build the fused belief structure then apply the evidence discount.

One may easily show that the FC configuration feature produces the following belief structure before discounting evidence:

$$
\begin{aligned}
&\text{For any E in } \Omega, \text{ we have:} \\
&m(E) = m_1 \oplus \ldots \oplus m_n (E) = \alpha \textstyle\sum_{E_1,\ldots E_q \leq \Omega; E_1 \wedge \ldots \wedge E_n = E} \prod_{i=1,n} m_i(E_i), \qquad (6) \\
&\text{where:} \\
&\alpha^{-1} = 1 - \textstyle\sum_{E_1,\ldots E_q \leq \Omega; E_1 \wedge \ldots \wedge E_n = \emptyset} \prod_{i=1,n} m_i(E_i).
\end{aligned}
$$

One may also easily show that the FD configuration feature produces the following belief structure before discounting evidence:

$$
\begin{aligned}
&\text{For any E in } \Omega: \\
&m(E) = m_1 \vee \ldots \vee m_n (E) = \textstyle\sum_{E_1,\ldots E_n \leq \Omega; E_1 \vee \ldots \vee E_n = E} \prod_{i=1,n} m_i(E_i) \qquad (7)
\end{aligned}
$$

Independently of the sensor configuration features selected, we obtain the following belief structure, after applying evidence discount,:

$$
\begin{aligned}
&m^{\delta}(E) = \delta m(E) \text{ for any E in } \Omega; \\
&m^{\delta}(\Omega) = (1-\delta) m(\Omega). \qquad (9)
\end{aligned}
$$

The pignistic model:

Even though we herein demonstrate the pignistic model, the interested reader may alternatively choose to compute Shafer's plausibility functions as a substitute to the pignistic probabilities.

Smets' pignistic probabilities may be induced from the above belief function as follows:

For any F in Ω,
$p(F) = \sum_{E \leq \Omega} m^{\delta}(E)|F \Lambda E|/|E|$. (10)

We just showed how to use the TBM to travel from the initial specifications defined in the corporate security policy to the design of an idrs generator's fuser capable of incorporating major sensors' configurations while incorporating Shafer's evidence discounts expressing sensors' reliability conditions. The final fused message produced by the fuser will be transferred to the idrs generator's classifier.

As an example, assume that our idrs employs 5 network sensors, as shown in Figure 3, placed at: net1, net2, firewall1, router1, and router2. Also assume that we are only concerned with 3 features: F1, F2, and F3, that take values in {L, M, H}, and 3 types of intrusions: i1, i2, and i3. Table 3 provides sensors reports. Table 4 gives the results of the fusion of sensors' reports.

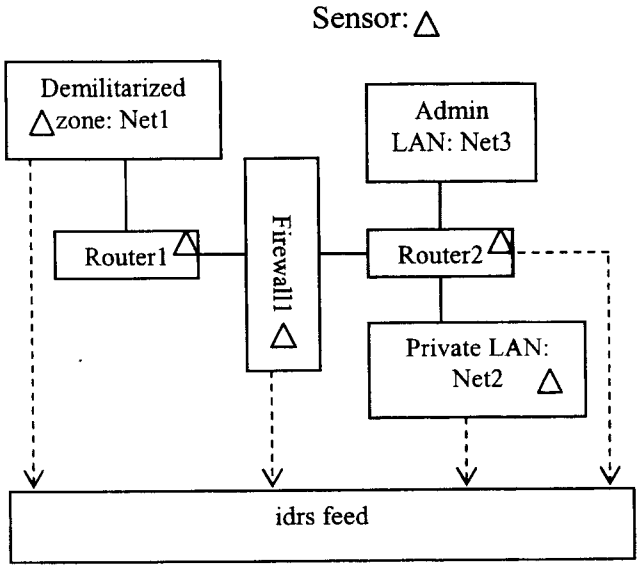


Figure 3: an illustration of a fuser

Table 3: Sensors reports and belief structure				
Objects	**F1**	**F2**	**F3**	**bel on Classes**
S1: in net1	L	M	M	m1:(i2: .3; Ω:.7)
S2: in firewall1	M	H	L	m2:(i1: .2; Ω:.8)
S3: router1	L	M	M	m3:(i3:.4; Ω:.6)
S4: net2	H	M	H	m4: (Ω :1.0)
S5: router2	L	H	H	m5: (i2vi3:.2; Ω:.8)

Table 4: Fusion of Sensors' reports								
	Ø	**Ω**	**i1**	**i2**	**i3**	**i1vi2**	**i1vi3**	**i2vi3**
Bel fused	.00	.98	.09	.17	.27	.26	.36	.44
m fused	.00	.34	.09	.17	.27	.00	.00	.09

The classifier

The corporate security policy related to intrusion detection should indicate whether we are allowed to design a supervised classifier or an unsupervised learner. Some information owners do not allow unsupervised learning because they are very risk averse to all simulation techniques including random sampling used in machine learning and in the statistics community. Other information owners may not approve supervised learning when they are not sure of the quality of the training data sets. Anyway, classifiers may be designed to provide supervised learning provided that there are sufficient cases for training and also sufficient cases for testing and for preventing over fitting.

Some of the techniques that a classifier can adopt for its design include: genetic computing, neural computing, Bayesian reasoning, Dempster and Shafer theory, Fuzzy set theory, Fuzzy expert system, decision tree, etc.

This area of research has enjoyed great coverage in the literature, and is still very popular in the data mining community. While we herein discuss the design specifications of a belief tree classifier, you may alternatively select any model of your choice.

Classification is an important decision support aid [1]. Diverse classification models have been proposed in the literature [25, 35]. Decision trees are attractive for their intuitive representation, easy assimilation [6], their cost-effectiveness [23], and their precision superiority [16, 20]. Within the area of decision tree classification, there are many algorithms to construct decision trees; you may just choose one of your choices to incorporate in the idsr generator. Popular decision tree algorithms reported in the literature and addressed extensively in statistics and machine learning include C4.5 [17], CART [6], CHAID [22], FACT [21], ID3 and extensions [7, 14, 27, 28, 29], SLIQ and Sprint [23, 24] and QUEST [20].

A simple decision tree growing process may look as follows:

1. *D=D1 v D2 (D1, for training, and D2 for testing and preventing over fitting)*
2. *Identify the attribute with the largest incremental gain in Shannon's information, given an initial partition of the training data set, obtained from a discounted belief structure*
3. *Reiterate with 2 until you have no longer new attributes*
4. *Test for over fitting by applying the testing data set D2. Use any pruning method of your choice to improve the belief decision tree.*

For the design of our idrs system, we simply grow the decision tree using the information gain concept applied to pignistic probabilities obtained from belief functions. That is, the current attribute to be selected is the attribute that maximizes information gain on the belief structure given the current training data set partition. Alternatively, one may choose to use plausibility functions instead of pignistic probabilities. The tree growing steps reiterate until we are out of attributes. Once we are done, over fitting may be eliminated with post-pruning using the testing data set.

While most ids systems reported in the literature use training data associated with known attack classes, our idrs system employs a training data set where the classes are not known for certain since they are expressed using Shafer's belief functions. Each class c_i is however defined by a bba m_i. The training data set may be defined using an induced belief decision tree.

Let A be an attribute taking values in $\{A_1, \ldots, A_m\}$ based on which a training data set D may be reorganized into a partition P(A) with k subsets $\{P_1(A), \ldots, P_k(A)\}$. Given n classes $\{C_1, \ldots, C_n\}$ constituting our frame of discernment, the incremental gain g(A) of information credited to the attribute A is defined by entropy reduction as:

$$g(A) = i(\emptyset) - i(A)$$
$$i(\emptyset) = e(D) = -p_D(c)\sum_c lb(p_D(c)) \qquad (11)$$
$$i(A) = e(A) = -\sum_a p_a(c)\sum_c lb(p_a(c))$$

Where:

a= values in $\{A_1, \ldots, A_m\}$,
p_D= pignistic probabilities associated with D
p_a= pignistic probabilities associated with a
lb: binary logarithm.

The term e(P) denotes the average amount of information needed to classify a specified case in a given partition P of the training data set D. Given the state of the tree being grown, the attribute having the highest information gain will be selected to grow the tree at the current node.

Here is an example: Assume that we are concerned with three intrusion attributes of features defining probe/scanning activities, system performance, and failed attempts of unauthorized access as follows:

Scan(S)={L: low, H: high}
Performance (P)={L: low, H: high}
Unauthorized access attempts (UAA)={L: low, H: high}

Also assume that we are only concerned with three classes of intrusions defined as follows:

Classes: C1= Information Leakage
C2= Information Corruption
C3= Denial of service

Let the basic belief assignments for the training data set D, provided in Table 5, be defined as follows:

m1(C1)=0.4; m1(C1vC2)=0.5; m1(Ω)= 0.1
m2(C1)=0.5; m2(C1vC3)=0.2; m2(Ω)= 0..3
m3(C2)=0.7; m3(Ω)= 0.3
m4(C3)=0.2; m4(C2vC3)=0.5; m4(Ω)= 0.3
m5(C2)=0.6; m5(Ω)= 0.4
m6(C1)=0.2; m6(C3)=0.4; m6(Ω)= 0.4

Table 5: Training data

Case	S	P	UAA	Classes
1	L	L	L	m1
2	L	H	L	m2
3	H	L	H	m3
4	H	H	L	m4
5	L	L	H	m5
6	H	L	H	m6

Table 6: Computation of Bel and m on the data set D

	Ø	Ω	C1	C2	C3	C1vC2	C1vC3	C2vC3
bel(D)	.00	.91	.18	.21	.03	.39	.21	.24
m_D	.00	.30	.18	.21	.03	.08	.03	.08

P.S. Note, in most tables of this article, the bbas and probabilities do not add to 1; simply because they are computed by hand and rounded inconsistently.

Table 7: Pignistic probabilities associated with m_D			
	C1	C2	C3
m_D	0.33	.39	.18
Entropy e(D) = 1.502927			

Table 8: Computation of Bel and m for the new tree								
	Ø	**Ω**	**C1**	**C2**	**C3**	**C1vC2**	**C1vC3**	**C2vC3**
Low scan	.00	.26	.30	.20	.00	.16	.06	0.00
High scan	.00	.33	.06	.23	.13	.00	.00	.16
Low per	.00	.30	.15	.32	.10	.12	.00	.00
High per	.00	.30	.25	.00	.10	.00	.10	.25
Low UAA	.00	.23	.30	.00	.06	0.16	.06	.16
High UAA	.00	.36	.06	.43	.13	.00	.00	.00

Table 9: Belief functions for intrusion attributes								
	Ø	**Ω**	**C1**	**C2**	**C3**	**C1vC2**	**C1vC3**	**C2vC3**
Bel: Low scan	.00	.98	.30	.20	.00	.50	.30	.20
Bel: High scan	.00	.91	.06	.23	.13	.29	.19	.36
Bel: Low per	.00	.99	.15	.32	.10	.47.	.25	.42
Bel: High per	.00	1.0	.25	.00	.10	.25	.35	.10
Bel: Low UAA	.00	.97	.30	.00	.06	.30	.36	.06
Bel: High UAA	.00	.98	.06	.43	.13	.49	.19	.56

Table 10: Pignistic Probabilities

	C1	C2	C3
Pignistic: Low scan	.49	.36	.11
Pignistic: High scan	.17	.42	.32
Pignistic: Low per	.31	.48	.20
Pignistic: High per	.40	.22	.37
Pignistic: Low UAA	.48	.23	.24
Pignistic: High UAA	.18	.55	.25

Most data and information needed to demonstrate an example, for the design of the fuser, are available in Tables 5 to 10.

Shannon's information gain from the belief structure may be computed as follows:

e(scan) = 1.435725
e(performance)= 1.518261
e(unauthorized access attempts)= 1.454876
e(D) = 1.502927

g(scan) = e(D) - e(scan) = 0.067202
g(performance) = e(D) - e(performance) = -0.015334
g(unauthorized access attempts)= e(D) - e(unauthorized access attempts)
= 0.048051

Just to show how Shannon's information gain on the belief structure is computed, we show the computation for the scan attribute:

$$e(\text{Scan}) = e(\text{Low scan}) + e(\text{High scan})$$
$$= -3/6 \sum_{(i=1,3)} p_L(C_i)\text{lb}(p_L(C_i) - 3/6 \sum_{(i=1,3)} [p_H(c_i)\text{lb}(p_H(c_i))$$
$$= -.5[.49*\text{lb}(.49) + .36\text{lb}(.36) + .11(\text{lb}(.11)]$$
$$-.5[.17*\text{lb}(.17) + .42(\text{lb}(.42) + .32\text{lb}(.32)]$$
$$=1.435725$$

where p_L and p_H denote the pignistic probabilities of low and high scanning activities.

Since there is higher gain of information with the scan attribute, this attribute should be selected to be the root of the tree. The initial set D will be then divided into two subsets, one corresponds to low scanning activities and the other to high scanning activities. We obtain a partition with two subsets D1 and D2, as shown in Figure 4. We reiterate with the same process applied independently to D1 and D2. The growing process stops when you encounter a leaf, a singleton partition subset.

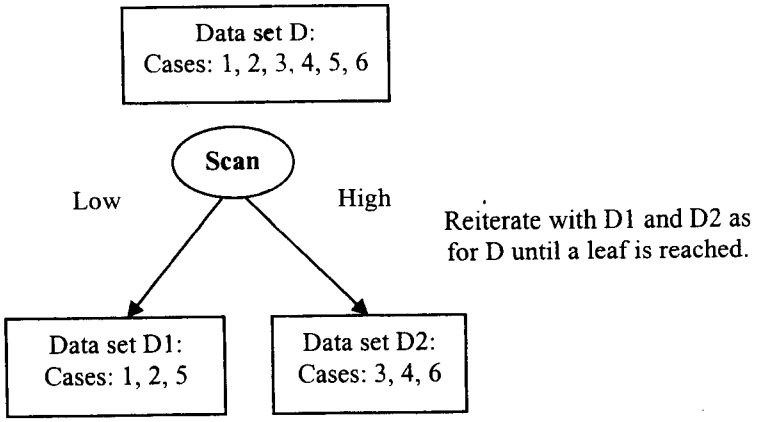


Figure 4: Growing the belief tree

The incident responder

The responder, as shown in Figure 5, fits the specifications of a Mamdani's fuzzy rule base system (MFRBS), for the fuser produces a basic belief assignment that can be easily transformed into a fuzzy subset [2, 11]. In fact, you also can skip the computation of pignistic probabilities and of Shafer's plausibility functions, as the security officer may not need to interpret the classifier output but instead wait for the recommendations generated by the responder.

In order to produce a highly descriptive model of the computing environment, and achieve easy interpretability of the responder output, a MFRBS will produce rules defining system behavior as a conjunction of linguistic terms and their labels. This will allow for a more global and an easier interpretation of system statements detailed in the corporate security policy.

The literature contains a decent amount of studies on FRBSs [3, 8, 9, 10, 13, 18]. The closest to what we are doing here will be Duns [3, 13] and Chiu [8] who applied fuzzy clustering techniques that derive partitions of the input and output fuzzy variables needed to produce fuzzy rules. Their learning process generates fuzzy rules using clusters centers. Herrera et al. [18] and Herrera [9, 10] have adopted genetic learning for approximative FRBS where the learning process uses an optimization problem for which a genetic algorithm is used to search for the best individual rules that optimise a prescribed objective function.

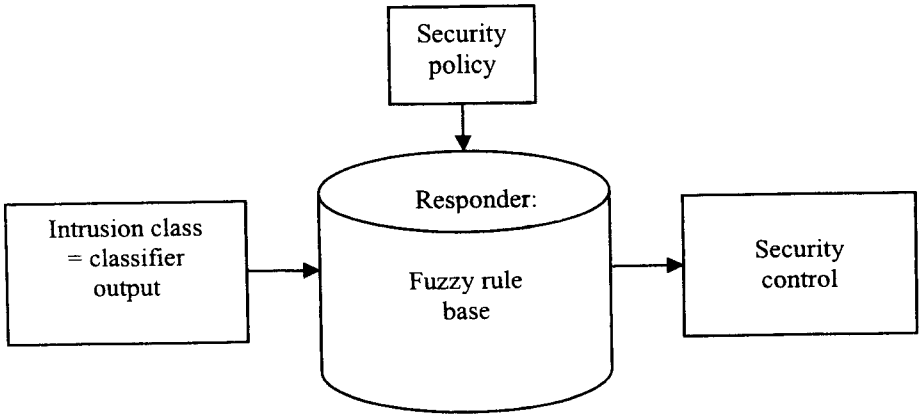


Figure 5: Responder specs

Even though intrusion detection can take different approaches, they all aim at identifying events of unauthorized access or penetration to the firm's computing environment. The bottom line should be the detection of all violations of the corporate security policy since a security policy is the definition of the acceptable behavior of the firm's computing environment. A simple form of intrusion detection may aim at looking for activities that are different from the user's or systems normal behavior.

In fact, for the idrs rule base, there is not really any difference between a misuse, an anomaly, or a policy approach, as long as content is represented in fuzzy rules. The same specifications apply to all intrusion detection and response approaches. In this case, the specifications of a Mamdani fuzzy rule base system should apply. Given an intrusion type, the responder searches the rule base for the most appropriate security control. A security control may consist of any action, device, procedure, technique, or other measure that reduces the vulnerability of a component of the computing environment.

Conclusion

This article discussed the design of an intrusion detection and response system generator. The design of the idrs generator included an evidence fuser, a classifier, and an incident responder. This system was designed to accept three main input streams: the firm's security policy, its current risk profile, and training data sets, and to produce an incident response in terms of managerial, technical, and operational security controls that security officers feasibly adopt to improve the firm's risk position as indicated in the corporate security policy. This article did not present a prototype of the idrs generator but demonstrated sufficient details about the use of the Transferred Belief Model in both the fuser and the classifier supported by Smets' pignistic probabilities.

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Database mining: A performance perspective. *IEEE TKDE*, December 1993.

[2] R. Alcala, J. Casillas, O. Cordón, F. Herrera, Approximate Mamdani-type Fuzzy Rule-Based Systems: Features and Taxonomy of Learning Methods, *Technical Reports #DECS AI-990117* (1999).

[3] J.C. Bezdek, Pattern *recognition with fuzzy objective function algorithms* (Plenum Press, 1981).

[4] L. Breiman, J.H. Friedman, R.A. Olshen, & P.J. Stone, *Classification and Regression Trees,* Wadsworth, Belmont, CA, 1984.

[5] L. Breiman, Bagging Predictors, *Machine Learning*, vol. 24, pp. 123-140, 1996.

[6] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. decision tree pruning. In *Proc. of KDD*, 1995.

[7] J.. Cheng, U.M. Fayyad, K.B. Irani, and Z. Qian. Improved decision trees: A generalized version of ID3. *Proc. of the 5th International Conference on Machine Learning*, San Mateo, CA: Morgan Kaufman, 100-106.

[8] S.L. Chiu, Fuzzy model identification based on cluster estimation, *Journal of Intelligent and Fuzzy Systems* 2:267-278 (1994).

[9] O. Cordón, F. Herrera, A three-stage evolutionary process for learning descriptive and approximate fuzzy logic controller knowledge bases from examples, *Int. Journal of Approximate Reasoning* 17(4): 369-407 (1997).

[10] O. Cordón, F. Herrera, Hybridising genetic algorithms with sharing scheme and evolution strategies for designing approximate fuzzy rule-based systems, *Fuzzy Sets and Systems* (1999).

[11] O. Cordón, F. Herrera, L. Magdalena, P. Villar. A Genetic Learning Process for the Scaling Factors, Granularity and Contexts of the Fuzzy Rule-Based System Data Base. *Information Science* 136 (2001) 85-107.

[12] Dethy, Examining port Scan Methods- Analysing Auditable Techniques, *white papers, tsr* 2001.

[13] J.C. Dunn, A fuzzy relative of the ISODATA process and its use in detecting compact well separated clusters, *Journal Cybernetics* 3:3 (1974) 32-57.

[14] U.M. Fayyad. *On the induction of decision trees for multiple concept learning*. PhD thesis, EECS Department, The University of Michigan, 1991.

[15] A. González and R. Pérez. Completeness and consistency conditions for learning fuzzy rules. *Fuzzy Sets and Systems*, 96:37-51, 1998.

[16] D.J. Hand. *Construction and Assessment of Classifi-cation Rules*, 1997.

[17] F. Herrera, M. Lozano, J.L. Verdegay, *Generating Fuzzy Rules From Examples using Genetic Algorithms,* 1995.

[18] F. Herrera, M. Lozano, J.L. Verdegay, A learning process for fuzzy control rules using genetic algorithms, *Fuzzy Sets and Systems* 100 (1998) 143-158.

[19] H. Ishibuchi, T. Murata, and I. B Turksen. Single-objective and two-Objective genetic algorithms for selecting linguistic rules for pattern classification problems. *Fuzzy Sets and Systems*, 89:135-150, 1997.

[20] T.-S. Lim, W.-Y. Loh, and Y.-S. Shih. *An empirical comparison of decision trees and other classification methods*. TR 979, Department of Statistics, UW Madison, June 1997.

[21]W.-Y. Loh and N. Vanichsetakul. Tree-structured classification via generalized disriminant analysis. *Journal of the American Statistical Association*, 83:715–728, 1988.

[22] J. Magidson. The CHAID approach to segmentation modeling. In *Handbook of Marketing Research*, 1993.

[23] M. Mehta, R. Agrawal, and J. Rissanen. SLIQ: A fast scalable classifier for data mining. In *Proc. of EDBT*, 1996.

[24] M. Mehta, J. Rissanen, and R. Agrawal. MDL-based decision tree pruning. In *Proc. of KDD*, 1995.

[25] D. Michie, D.J. Spiegelhalter, and C.C. Taylor, editors. *Machine Learning, Neural and Statistical Classification*, 1994.

[26] M. Phayung, Quantitative Measures of a Fuzzy Expert System, *Final Report IEEE NNC Student Summer Research* 2001.

[27] J.R. Quinlan. Discovering rules by induction from large collections of examples. In *Expert Systems in the Micro Electronic Age*, 1979.

[28] J.R Quinlan. Learning efficient classification procedures. In *Machine Learning: An Artificial Intelligence Approach*, 1983.

[29] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[30] J.R. Quinlan. *C4.5: Programs for Machine Learning*, 1993.

[31] P. Smets. (1993). Belief functions: the disjunctive rule of combination and the generalized Bayesian theorem. *International Journal of Approximate Reasoning*, 9, 1-35.

[32] P. Smets & Kennes, R. (1994). The transferable belief model. Artificial Intelligence, 66, 191-234.

[33] P. Smets (1998). The transferable belief model for quantified belief representation. In D. M. Gabbay & P. Smets (Eds.), *Handbook of defeasible reasoning and uncertainty management systems*, vol. 1 (pp. 267-301). Doordrecht, The Netherlands: Kluwer.

[34] P. Smets (1997). The normative representation of quantified beliefs by belief functions. *Artificial Intelligence*, 92, 229-242.

[35] S.M.Weiss and C.A. Kulikowski. *Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems*, 1991.

School of Computer Science and Information Systems
Pace University
Technical Report Series

EDITORIAL BOARD

Editor:
Allen Stix, Computer Science, Pace--Westchester

Associate Editors:
Constance A. Knapp, Information Systems, Pace--New York
Susan M. Merritt, Dean, SCSIS--Pace

Members:
Howard S. Blum, Computer Science, Pace--New York
Mary F. Courtney, Computer Science, Pace--Westchester
Nicholas J. DeLillo, Mathematics and Computer Science, Manhattan College
Fred Grossman, Information Systems; Doctor of Professional Studies, Pace--New York and White Plains
Fran Goertzel Gustavson, Information Systems, Pace--Westchester
Joseph F. Malerba, Computer Science, Pace--Westchester
John S. Mallozzi, Computer Information Sciences, Iona College
John C. Molluzzo, Information Systems, Pace--New York
Pauline Mosley, Technology Systems, Pace--New York
Narayan S. Murthy, Computer Science, Pace--New York
Catherine Ricardo, Computer Information Sciences, Iona College
Judith E. Sullivan, CSIS Research and Assessment; Technology Systems, Pace--Westchester
Sylvester Tuohy, Computer Science, Pace--Westchester

The School of Computer Science and Information Systems, through the Technical Report Series, provides members of the community an opportunity to disseminate the results of their research by publishing monographs, working papers, and tutorials. *Technical Reports* is a place where scholarly striving is respected.

All preprints and recent reprints are requested and accepted. New manuscripts are read by two members of the editorial board; the editor decides upon publication. Authors, please note that production is Xerographic from the pages you have submitted. Statements of policy and mission may be found in issues #29 (April 1990) and #34 (September 1990).

Please direct submissions as well as requests for single copies to:

Allen Stix
School of CS & IS - Goldstein Academic Center
Pace University
861 Bedford Road
Pleasantville, NY 10570-2799