Network Intrusion Detection Using Tree Augmented Naive-Bayes

R. Najafi [*]
Sanay Systems
najafirobab88@gmail.com

Mohsen Afsharchi
University of Zanjan
afsharchim@znu.ac.ir

Abstract: Computer networks are nowadays subject to an increasing number of attacks. Intrusion Detection Systems (IDS) are designed to protect them by identifying malicious behaviors or improper uses. Since the scope is different in each case (register already-known menaces to later recognize them or model legitimate uses to trigger when a variation is detected), IDS have failed so far to respond against both kind of attacks. In this paper, we apply two of the efficient data mining algorithms called Naive Bayes and tree augmented Naive Bayes for network intrusion detection and compare them with decision tree and support vector machine. We present experimental results on NSL-KDD data set and then observe that our intrusion detection system has higher detection rate and lower false positive rate.

Keywords: Anomaly And Misuse Detection, Bayesian Network, Intrusion detection, Tree Augmented Naive-Bayes, Naive-Bayes.

1 Introduction

Intrusion detection techniques are the last line of defense against computer attacks behind secure network architecture design, firewalls, and personal screening. Despite the plethora of intrusion prevention techniques available, attacks against computer systems are still successful. Thus, intrusion detection systems (IDSs) play a vital role in network security. The attacks are targeted at stealing confidential information such as credit card numbers, passwords, and other financial information. One solution to this dilemma is the use of intrusion detection system (IDS). It is very popular security tool over the last two decades, and today, IDS based on computer intelligent are attracting attention of current research community a lot.

We present experimental results on NSL-KDD data set and WEKA software.

Valdes and Skinner employed a naive Bayesian network to perform intrusion detection on network events. The classification capability of a naive Bayesian network is identical to a threshold-based system that computes the sum of the outputs obtained from the child nodes. This is due to the fact that all models (i.e., the child nodes) operate independently and only influence the probability of the root node. This single probability value at the root node can be represented by a threshold. In addition, the restriction of having a single parent node complicates the incorporation of additional information. This is because variables that represent such information cannot be linked directly to the nodes representing the model outputs. [2]

ADAM (Audit Data Analysis and Mining) is an intrusion detector built to detect intrusions using data mining techniques. It first absorbs training data known to be free of attacks. Next, it uses an algorithm to group attacks, unknown behaviors, and false alarms. ADAM has several useful capabilities, namely;

- Classifying an item as a known attack
- Classifying an item as a normal event
- Classifying an item as an unknown attack
- Match audit trial data to the rules it gives rise to. [8]

Also, TAN algorithm can be used for ranking, regres-

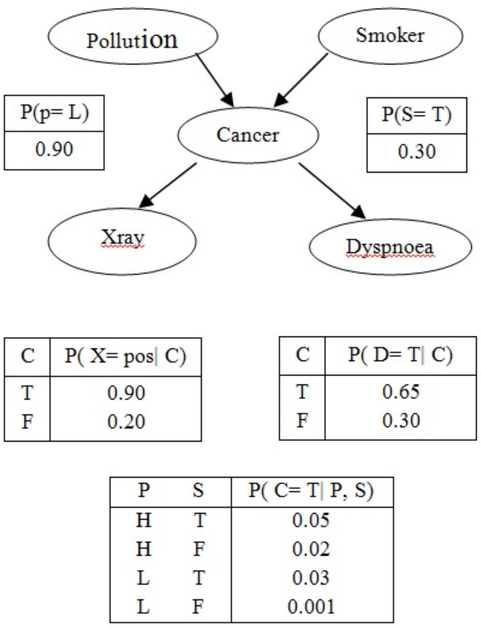
[*]Corresponding Author, P. O. Box 45195-1159, F: (+98) 241 421-5071, T: (+98) 241 415-5067

sion analysis, probability estimation and engine fault diagnosis improvement.

The paper is structured as follows: Section 2 introduces Bayesian networks and classifications. Section 3 introduces intrusion detection systems and different kinds of attacks. Section 4 describes intrusion detection with Bayesian networks. Section 5 presents and analyzes our experimental results. Section 6 summarizes the main conclusions.

2 Primary Description

A Bayesian network $B =< N, A, \Theta >$ is a directed acyclic graph (DAG) $< N, A >$ where each node $n \in N$ represents a domain variable (e.g., a dataset attribute), and each arc $a \in A$ between nodes represents a probabilistic dependency, quantified using a conditional probability distribution (CP table) $\theta_i \in \Theta$ for each node n_i. A BN can be used to compute the conditional probability of one node, given values assigned to the other nodes [3].



P(p= L)
0.90

P(S= T)
0.30

C	P(X= pos| C)
T	0.90
F	0.20

C	P(D= T| C)
T	0.65
F	0.30

P	S	P(C= T| P, S)
H	T	0.05
H	F	0.02
L	T	0.03
L	F	0.001

Figure 1: Bayesian network for cancer

The main advantages of Bayesian networks are:

- Bayesian networks can readily handle incomplete data sets.
- Bayesian networks allow one to learn about causal relationships.
- Bayesian networks in conjunction with Bayesian methods and other types of models offer an efficient and principled approach for avoiding the overfitting of data. [1]

Bayesian network structure represents the inter-relationships among the dataset attributes. Human experts can easily understand the network structures and if necessary modify them to obtain better predictive models. By adding decision nodes and utility nodes, BN models can also be extended to *decision networks* for decision analysis. Applying Bayesian network techniques to classification involves two subtasks: BN learning (training) to get a model and BN inference to classify instances. The two major tasks in learning a BN are: learning the graphical structure, and then learning the parameters (CP table entries) for that structure.

The set of parents of a node x_i in B_S is denoted as π_i. The structure is annotated with a set of conditional probabilities (B_P), containing a term $P(x_i = X_i|\pi_i = \Pi_i)$ for each possible value Xi of xi and each possible instantiation Π_i of π_i. [3]

One application of Bayesian networks is classification. A somewhat simplified statement of the problem of supervised learning is as follows. Given a *training set* of labeled instances of the form $< a_1, ..., a_n >$, C construct a *classifier f* capable of predicting the value of C, given instances $< a_1, ..., a_n >$ as input. The variables $A_1, ..., A_n$ are called *features* or *attributes*, and the variable C is usually referred to as the *class variable* or *label.* [11]

Two types of Bayesian network classifiers that we use them in this paper are: Naive-Bayes and Tree Augmented Naive-Bayes

2.1 Naive-Bayes

A Naive-Bayes BN is a simple structure that has the class node as the parent node of all other nodes (see Figure 1). No other connections are allowed in a Naive-Bayes structure. Naive-Bayes assumes that all the features are independent of each other. In recent years, a lot of effort has focused on improving Naive-Bayesian classifiers, following two general approaches: selecting feature subset and relaxing independence assumptions.

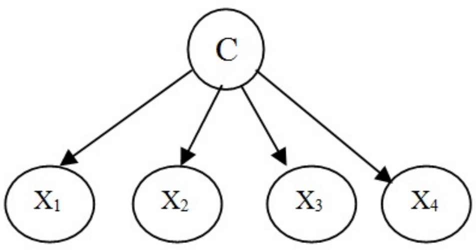


Figure 2: Naive-Bayes Structure [6]

2.2 Tree Augmented Naive-Bayes (TAN)

TAN classifiers extend Naive-Bayes by allowing the attributes to form a tree, (see Figure 2) here c is the class node, and the features x_1, x_2, x_3, x_4, without their respective arcs from c, form a tree. [6]

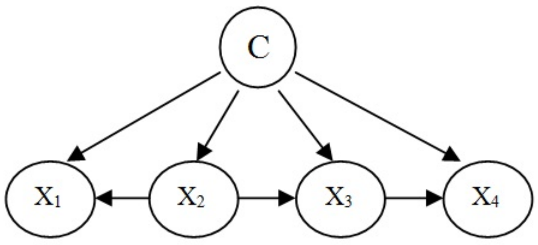


Figure 3: TAN Structure [6]

3 Intrusion Detection Systems

Intrusion detection systems are used to identify, classify and possibly, to respond to benign activities. Also, Intrusion Detection System (IDS) is used to monitor all or partial traffic, detect malicious activities, and respond to the activities. Network intrusion detection system was established for the purpose of malicious activities detection to strengthen the security, confidentiality, and integrity of critical information systems. These systems can be network-based or host-based. HIDS is used to analyze the internal event such as process identifier while NIDS is to analyze the external event such as traffic volume, IP address, service port and others. The challenge of the study is: how we can have an IDS with high detection and low false positive rate? [4]

Intrusion detection is comprised of two main techniques which are misuse-based intrusion detection and anomaly based intrusion detection.
Misuse-based intrusion detection IDSs that employ misuse detection approach detect attacks by comparing the existing signatures against the network traffics captured by the IDSs. When a match is found, the IDSs will take action as the traffics are considered harmful to computer systems or computer networks. Actions taken by the IDSs will normally include sending alerts to network administrator and logging the intrusive events.

IDSs that implement misuse detection approach are, however, incapable of detecting novel attacks. The network administrator will need to update the stored signatures frequently to make sure that the IDSs perform well in detecting intrusions. [5]
Anomaly based intrusion detection IDSs that employ anomaly detection are capable of identifying novel attacks, that contain activities deviated from the norm. Such IDSs utilize the built profiles that are learned based on normal activities in computer networks. This system has two stages:

- **Learning:** It works on profiles. The profiles represent the normal behavioural activities of the users, systems, or network connections, applications. Great care should be taken while defining profiles because currently there is no effective way to define normal profiles that can achieve high detection rate and low false positives at the same time.
- **Detection:** The profile is used to detect any deviance in user behavior. [7]

3.1 Problems of Intrusion Detection Systems

IDS have three common problems: *temporal complexity, correctness and adaptability.*
The temporal complexity problem results from the extensive quantity of data that the system must supervise in order to perceive the whole situation. False positive and false negative rates are usually used to evaluate the correctness of IDS. False positive can be defined as alarms which are triggered from legitimate activities. False negative includes attacks which are not detected by the system. An IDS is more precise if it detects more attacks and gives few false alarms.

In case of misuse detection systems, security experts must examine new attacks to add their corresponding signatures. In anomaly detection systems, human experts are necessary to define relevant attribute for defining the normal behavior. This leads us to the adaptability problem. [10]

3.2 Different Types of Attacks

Attacks are grouped into four classes:

- **Denial of Service (DOS):** Making some machine resources unavailable or too busy to answer to legitimate users requests.
- **User to Root (U2R):** Exploiting vulnerability on a system to obtain a root access.
- **Remote To Local (R2L):** Using vulnerability in order to obtain a local access like a machine user.
- **Probing:** Collecting useful information or known vulnerabilities about a network or a system. [8]

4 Network Intrusion Detection Using Bayesian Networks

In the following we will first discuss the Naive-Bayes and then explore our contribution which is Tree Augmented Naive-Bayes in intrusion detection.

4.1 Naive-Bayes

The purpose is to find the probability that a computer or local network attack is going on.The result of the propagation of changed probabilities of certain events observed by Bayesian network can be an automatic activation of some mechanism for attack prevention such as: breaking TCP connections, traffic redirection or disabling user activity. If the probability of an attack is significantly increased but not enough to be considered as an attack, the network will generate a report about the event and warn the system administrator. Once the network is quantified, it is able to classify any new object giving its attributes value using the Bayes rule expressed by:

$$P(C_i|A) = \frac{P(A|C_i)P(C_i)}{P(A)} \tag{1}$$

Where c_i is a possible value in the session class and A is the total evidence on attributes nodes. The evidence A can be dispatched in the pieces of evidence, say $a_1, a_2, ..., a_n$ relative to the attributes $A_1, A_2, ..., A_n$, respectively. Since naive Bayesian networks work under the assumption that these attributes are independent (giving the parent node C), their combined probability is obtained as follows:

$$P(C_i|A) = \frac{P(a_1|C_i)P(a_2|C_i)...P(a_n|C_i)P(C_i)}{P(A)} \tag{2}$$

Note that there is no need to explicitly compute the denominator P(A) since it is determined by the normalization condition. Therefore, it is sufficient to compute for each class c_i its likelihood, i.e. $P(a_1|C_i)P(a_2|C_i)...P(a_n|C_i)P(C_i)$ to classify any new object characterized by its attributes values $a_1, a_2, ..., a_n$. [9]

4.2 Tree Augmented Naive-Bayes

Bayesian network structure learning without any structural restrictions is known to be a difficult problem. Several possibilities of adding arcs between classifier features have been proposed. TAN models are well-known extensions of naive Bayes. The main rule of TAN classification is given by:

$$P(C|A) = \alpha P(C)P(A_1|PA_1, C)...P(A_n|PA_n, C) \tag{3}$$

A is the global information provided by features values. PA_i is the parent feature of A_i (on which A_i depends). The optimal tree can be obtained simply by calculating mutual information measures between each two variables on the basis of training instances. [10]

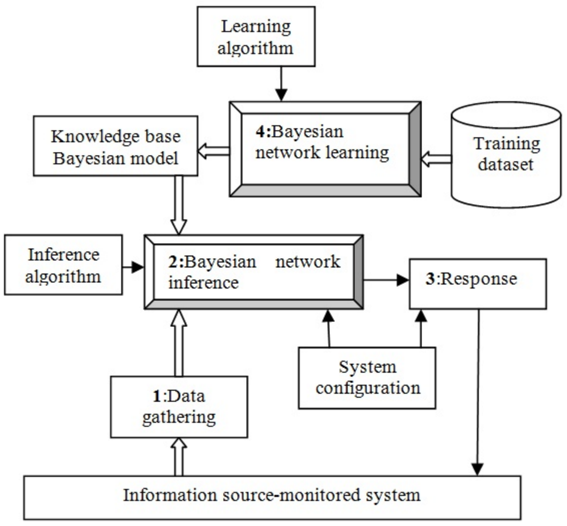

Learning algorithm
Knowledge base Bayesian model
4:Bayesian network learning
Training dataset
Inference algorithm
2:Bayesian network inference
3:Response
System configuration
1:Data gathering
Information source-monitored system

Figure 4: IDS with Bayesian Network

4.3 Algorithm

We present here an algorithm to achieve an optimal choice and placement of detectors.

Input

(i) Bayesian network BN = (V,CPT(V),H(V)) where V is the set of attack vertices, CPT(V) is the set of conditional probability tables associated with the attack vertices, and H(V) is the set of hosts affected if the attack vertex is achieved.

(ii) Set of detectors $D = (d_i, V(d_i), CPT[i][j])$ where d_i is the i^{th} detector, $V(d_i)$ is the set of attack vertices that the detector d_i can be attached to (i.e., the detector can possibly detect those attack goals being achieved), and CPT[i][j] $\forall j \in V(di)$ is the CPT tables associated with detector i and attack vertex j.

Output:

Set of tuples $\theta = (di, \Pi_i)$ where di is the i^{th} detector selected and Π_i is the set of attack vertices that it is attached to.

```
DETECTOR-PLACEMENT (BN, D)
System-Cost= 0
Sort all $(d_i, a_j), a_j \in V(d_i)$, i by
BENEFIT$(d_i, a_j)$
Sorted list= L
Length(L)= N
for $i$=1 to $n$ do
    System-Cost= System-Cost + $Cost(d_i, a_j)$
    /* $Cost(d_i, a_j)$ can be in terms of economic
    cost, cost due to false alarms and missed
    alarms, etc. */
    if $System - Cost > Threshold\tau$ then
        break
    end
    if $d_i \in \theta$ then
        add $a_j to \Pi_i \in \theta$
    else
        add$(d_i, \Pi_i = a_j) to \theta$
    end
end
return $\theta$
```

The worst-case complexity of this algorithm is O(dv B(v,CPT(v)) + dv log(dv) + dv), where d is the number of detectors and v is the number of attack vertices. B(v,CPT(v)) is the cost of Bayesian inference on a BN with v nodes and CPT(v) defining the edges. The first term is due to calling Bayesian inference with up to d times v terms. The second term is the sorting cost and the third term is the cost of going through the for loop dv times. [13]

```
BENEFIT (d, a)
/* This is to calculate the benefit from attaching
detector d to attack vertex a */
Let the end attack vertices in the BN be
$F = f_i, i = 1, 2, ..., M$
for each fi, the following cost-benefit table exists
do
    Perform Bayesian Inference with d as the
    only detector in the network and connected
    to attack vertex a
    Calculate for each $f_i$, the precision and
    recall, call them, Precision$(f_i, d, a)$ ,
    Recall$(f_i, d, a)$ $System\,Benefit =$
    $\sum_{i=1}^{m} Benefit\, f_i(True\,Negative) \times$
    Precision$(f_i, \mathrm{d}, \mathrm{a})$ + Benefit$f_i(True\,Positive)$
    $\times$ Recall$(f_i, \mathrm{d}, \mathrm{a})$
end
return System-Benefit
```

5 Experimental Result

The data used in this paper are those proposed in the NSL-KDD for intrusion detection which are generally used for benchmarking intrusion detection problems. They set up an environment to collect TCP/IP dump raws from a host located on a simulated military network. Each TCP/IP connection is described by 41 features and labeled as either normal or as an attack.

We evaluate the performance of Naive-Bayes and then we convert that to tree augmented Naive-bayes. So the new system has better performance.

parent	**childs**
dst_bytes	num_compromised
srv_count	count, srv_diff_host_rate
hot	is_guest_login
src_bytes	wrong_fragment, flag
count	same_srv_rate

Table 1: Connections in TAN

At the end compare them with DT and SVM. We use full training set and 10- fold cross validation for the testing purposes. In 10-fold cross-validation, the available data is randomly divided in to 10 disjoint subsets of approximately equal sizes. One of the subsets is then used as the test set and the remaining 9 sets are used for building the classifier. The test set is then used to estimate the accuracy. This is done repeatedly 10 times so that each subset is used as a test subset once. The accuracy estimates is then the mean of the estimates for each of the classifiers. Cross-validation has been tested extensively and has been found to generally work well when sufficient data is available.

5.1 Kappa Statistic Rate

The kappa statistic measures the agreement of prediction with the true class 1.0 signifies complete agreement. This rate in Naive-Bayes is 0.759 and in DT is 0.989 and in SVM is 0.961 but TAN has better result, 0.988.

5.2 Confusion Matrix

A Confusion Matrix is sometimes used to represent the result of testing, as shown in Table 1.It is a two- dimensional table with a row and column for each class, each element of the matrix show the number of test examples for which the actual class is the row and the predicted class is the column. The Advantage of using this matrix is that it not only tells us how many got misclassified but also what misclassifications occurred.

False Positive	Normal	DOS	R2L	Probe	U2R
NB	0.037	0.021	0.02	0.06	0.069
TAN	0.009	0.003	0.001	0.002	0
DT	0.01	0.002	0.001	0.003	0
SVM	0.031	0.005	0.002	0.003	0

Table 2: False Positive Rate

5.3 Time Taken to Build Model

Naive-Bayes is build in 3.77 seconds, TAN in 20.09 seconds, DT in 36.86 seconds and SVM in 43.63 seconds. So Naive-Bayes is faster.

5.4 Percent of Correct Classification

PCC of Naive-Bayes is %85 and PCC of TAN is %99.3 and PCC of DT is %99.4 and PCC of SVM is %97.8.

	Normal	**DOS**	**R2L**	**Probe**	**U2R**
Normal	3474	106	162	455	589
NB	4759	7	8	12	0
TAN	4760	5	8	13	0
DT	4732	17	14	24	0
SVM					
R2l	0	0	63	9	4
NB	5	0	69	1	1
TAN	10	0	63	2	1
DT	31	1	41	1	0
SVM					
DOS	115	3176	1	26	15
NB	9	3321	0	3	0
TAN	7	3319	1	6	0
DT	65	3264	0	1	0
SVM					
Probe	40	15	10	718	17
NB	22	10	0	768	0
TAN	20	9	1	770	0
DT	32	8	0	764	0
SVM					
U2R	0	0	2	0	2
NB	2	0	2	0	0
TAN	4	0	0	0	0
DT	3	0	1	0	0
SVM					

Table 3: Experimental Result in Confusion Matrix

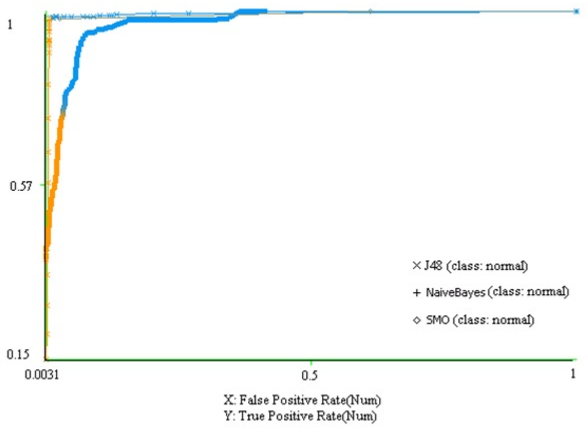


Figure 5: ROC Curve-Performance Analysis of IDS

5.5 Detection Rate and False Positive Rate

The detection rate is the number of attacks detected by the system divided by the number of attacks in the data set. It is equivalent to Recall.

The false positive rate is the number of normal connections that are misclassified as attacks divided by the number of normal connections in the data set.

The average of false positive rate in Naive-Bayes is 0.033 and in TAN and DT is 0.006, in SVM is 0.019.

5.6 Accuracy Rate

	Precision	Recall	F_Measure
Normal	0.957	0.726	0.826
NB	0.992	0.994	0.993
TAN	0.991	0.995	0.993
DT	0.973	0.989	0.981
SVM			
DOS	0.963	0.953	0.958
NB	0.995	0.996	0.996
TAN	0.996	0.996	0.996
DT	0.992	0.98	0.986
SVM			
R2l	0.265	0.829	0.401
NB	0.873	0.908	0.89
TAN	0.863	0.829	0.846
DT	0.732	0.554	0.631
SVM			
Probe	0.594	0.898	0.715
NB	0.98	0.96	0.97
TAN	0.973	0.963	0.968
DT	0.967	0.95	0.959
SVM			
U2R	0.003	0.5	0.006
NB	0	0	0
TAN	0	0	0
DT	0	0	0
SVM			
AVG	0.921	0.826	0.861
NB	0.991	0.991	0.991
TAN	0.99	0.99	0.99
DT	0.977	0.978	0.977
SVM			

Table 4: Accuracy rate in different algorithms

5.7 ROC Curve

We plot a ROC (Receive Operating Characteristic) curve which is often used to measure performance of IDS. This curve is a plot of the detection rate against the false positive rate, which is shown in Figure 5.

6 Conclusions

In this paper, we have proposed a framework of intrusion detection systems based on Naive-Bayes and TAN algorithms and compared them with decision tree and support vector machine. According to the result, Naive-Bayes is found less time consuming. TAN has better accuracy rate and detection rate, and also has less false positive rate.

Refrences

[1] K Korb and E Nicholson, *Bayesian Artificial Intelligence* (2004).

[2] Ch Kruegel, D Mutz, W Robrtson, and F Valeur, *Bayesian Event Classification For Intrusion Detection*, 19th Annual Computer Security Application Conference IEEE Computer Society, Washington DC **187** (2008), 14–23.

[3] L Ben-Gal, *Bayesian Network*, Encyclopedia Of Statistics In Quality And Reliability, 2007.

[4] Y Wee, W Cheah, SH Tan, and K Wee, *Causal Discovery And Reasoning For Intrusion Detection Using Bayesian Network* **1** (2011), no. 2.

[5] K Chin Khor, CH Ting, and S Amnuaisuk, *From Feature Selection To Building Of Bayesian Classifiers: A Network Intrusion Detection Perspective.*

[6] J Cheng and R Greiner, *From Feature Selection To Building Of Bayesian Classifiers: A Network Intrusion Detection Perspective*, Proc.14th Canadian Conference On AI, 2001.

[7] M Pater, H Kim, and A Pamnam, *State Of The Art In Intrusion Detection System.*

[8] M Panda and M.R Patra, *Network Intrusion Detection Using Nave Bayes*, International Journal Of Computer Science And Network Security **7** (2007), no. 12, 258–263.

[9] N Amor, S Benferhat, and Z Elovedi, *Nave Bayesian Networks In Intrusion Detection System*, 14th European Conference On Machine Learning 17th European Conference On Principles And Practice Of Knowledge Discovery In Databases.

[10] S Benferhat, H Drias, and A Boudjelida, *An Intrusion Detection Approach Based On Tree Augmented Nave-Bayes And Expert Knowledge.*

[11] N Friedman, M Goldzmidt, and A Boudjelida, *Building classifiers Using Bayesian Network*, In Proceeding of the National Conference on AI, Menlo Park, CA: AAAI Press (1996), 1277–1284.

[12] Cerquides Bueno J, *Improving Bayesian Network Classifiers*, Ph.D. Thesis, university Of Politecnica De Catalunya. (1996), 1277–1284.

[13] G Modelo-Howard, S Bagchi, and G Lebanon, *Determining Placement Of Intrusion Detectors For A Distributed Application Through Bayesian Network Modelling*, Springer-Verlage Berlin Heidelberg (2008), 271–290.