computing@tanet.edu.te.ua
www.computingonline.net

ISSN 1727-6209
International Journal
of Computing

K-MEANS FOR MODELLING AND DETECTING ANOMALOUS PROFILES

Rachid Beghdad

Faculty of sciences, 12 boulevard Bouaouina, Béjaïa 06000, Algeria.
e-mail: rbeghdad@yahoo.fr

Abstract: *We introduce an intrusion detection method based on the K-means (KM) clustering method to detect anomalous users' profiles. The main idea is to define k centroids, one for each cluster, such that each cluster represents a given user profile. These centroids should be placed as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early groupage is done. At this point we need to re-calculate k new centroids as barycenters of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new centroid. A loop has been generated. As a result of this loop we may notice that the k centroids change their location step by step until no more changes are done. An example and experiments are described to illustrate the robustness of our approach.*

Keywords: *Intrusion detection systems, Audit trail analysis, K-means, User behavior, Anomaly intrusion detection, Anomalous behavior.*

1. INTRODUCTION

An intrusion can be defined as a serie of activities aiming at compromising the security of a computer network system [1]. Intrusions may take many forms: external attacks, internal misuses, network-based attacks, information gathering, denial of service, and so on. Intrusion detection is an important step of protecting the computer network system from intrusions. Intrusion detection systems (IDS) are used to detect, identify and stop intruders.The administrators can rely on them to find out successful attacks and prevent a future use of known exploits. IDS are also considered as a complementary solution to firewall technology by recognizing attacks against the network that are missed by the firewall.

There are two basic types of intrusion detection: host-based and network-based. Each has a distinct approach to monitoring and securing data, and each has distinct advantages and disadvantages. In short, host-based IDSs examine data held on individual computers that serve as hosts, while network-based IDSs examine data exchanged between computers.

In addition to that, intrusion detection techniques can be mapped into four classes: anomaly detection, misuse detection, specification-based detection, and model-based detection. Anomaly detection, consists of establishing normal behavior profile for user and system activity and observing significant deviations of actual user activity with respect to the established habitual pattern. Misuse detection, refers to intrusions that follow well defined attack patterns that exploit weaknesses in system and application software. In specification-based detection, the correct behaviors of critical objects are manually abstracted and crafted as security specifications, which are compared with the actual behavior of the objects. Intrusions, which usually cause object to behavior in an incorrect manner, can be detected without exact knowledge about them. Model-based intrusion detection compares a process's execution against a program model to detect intrusion attempts.

We introduce here an anomaly intrusion detection method based on KM [2]. This method aims to find an optimum clustering (the best one) of users through a certain number of clusters *k* fixed a priori. At the end, if we find that some users are not well assigned according to this algorithm, then, we can conclude that they are suspicious (intruders).

The rest of this paper is organized as follows. Section 2 presents a survey of some intrusion detection methods. Our approach based on KM is detailed in section 3. An example is presented in section 4. Section 5 describes some experiments. Section 6 concludes the paper.

2. STATE OF ART

In this section, some intrusion detection models are presented.

2.1. ANOMALY DETECTION MODELS

Learning Vector Quantization network

In [3] the authors described some preliminary results concerning the robustness and generalization capabilities of machine learning methods in creating user profiles based on the selection and subsequent classification of command line arguments. They based their method on the belief that legitimate users can be classified into categories based on the percentage of commands they use in a specified period. The hybrid approach they employed begins with the application of expert rules to reduce the dimensionality of the data, followed by an initial clustering of the data and subsequent refinement of the cluster locations using a competitive network called Learning Vector Quantization (LVQ). Since LVQ is a nearest neighbor classifier, and new record presented to the network that lies outside a specified distance is classified as a masquerader. Thus, this system does not require anomalous records to be included in the training set.

Network-based Inrusion Detection Using NNs

The authors of [4] presented an anomaly detection system that detects network-based attacks by carefully analyzing this network traffic data and alerting administrators to abnormal traffic trends. It has been shown that network traffic can be efficiently modelled using artificial neural networks. Therefore they used MLP neural networks to examine network traffic data. In their system, it becomes necessary to group network traffic together to present it to the neural network. For this purpose, they used self-organizing maps, as they have been shown to be effective in novelty detection, automated clustering, and visual organization.

CDA model

We proposed in [5] a method based on Canonical Discriminant Analysis (CDA) for intrusion detection in computer systems. This approach leads to find intruders in a given computer system by making an appropriate discrimination between the users groups (user profiles).

Let Z a set of users (each user is defined by its profile vector) which is divided into k users groups ω_k of a given organization (university, enterprise, …). If we find that a user u does not belong to its initial group ω_i after the projection of its corresponding profile vector onto a discriminant plane, then this user will be suspicious (an intruder).

2.2. MISUSE DETECTION MODELS

eXpert-BSM

The "eXpert-Base Security Module" (eXpert-BSM) [6] is a real time forward-reasoning expert-system that analyzes Sun Solaris audit trails. eXpert-BSM's knowledge base detects a wide range of specific and general forms of misuse, provides detailed reports and recommendations to the system operator, and has a low false-alarm rate. Suites of eXpert-BSMs may be deployed throughout a network, and their alarms managed, correlated, and acted on by remote or local subscribing security services, thus helping to address issues of decentralized management. Inside the host, eXpert-BSM is intended to operate as a true security daemon for host systems, consuming few CPU cycles and very little memory and secondary storage, according to its authors.

CAML

The Correlated Attack Modeling Language (CAML) [7] uses a modular approach, where a module represents an inference step and modules can be linked together to detect multistep scenarios. CAML is accompanied by a library of predicates, which functions as a vocabulary to describe the properties of system states and events. The concept of attack patterns is introduced to facilitate reuse of generic modules in the attack modeling process. CAML is used in a prototype implementation of a scenario recognition engine that consumes first-level security alerts in real time and produces reports that identify multistep attack scenarios discovered in the alert stream.

PCA model

Shyu and al. [8] proposed a method that used principal component analysis (PCA) in intrusion detection problem where the training data may be unsupervised. Assuming that anomalies can be treated as outliers, an intrusion predictive model is constructed from the major and minor principal components of normal instances. A measure of the difference of an anomaly from the normal instance is the distance in the principal component space.

Chronicles model

In [9], the authors proposed a multi-alarm misuse correlation component based on the chronicles formalism. Chronicles provide a high level declarative language and a recognition system that is used in other areas where dynamic systems are monitored. This formalism allows them to reduce the number of alarms shipped to the operator and enhances the quality of the diagnosis provided. They illustrated how chronicles might solve some of current intrusion detection issues like alarm overload, false positives and poor alarm semantics.

Bayesian networks model

K. Johansen and al. [10] suggested a Bayesian system which would provide a solid mathematical foundation for simplifying a seemingly difficult and monstrous problem that today's Network Intrusion

Detection Systems (NIDS) fail to solve. The Bayesian NIDS (BNIDS) should have the capability to differentiate between attacks and the normal network activity by comparing metrics of each network traffic sample. Finally, such a NIDS should prove to be easily extendable and run in real-time while simple to maintain.

Decision trees model

T. Abbes and al [11] show that an arbitrary definition of the domain search for signatures causes the generation of false negatives.To overcome this problem, they rely on a protocol analysis approach that leads to the construction of decision trees in the initial phase of the IDS deployment.The built tree is adaptative to the network traffic characteristics since the features chosen to split the tree ensure the highest reduction of entropy or the lowest Gini impurity.In addition, pattern matching operations are integrated inside decision tree.They are triggered after achieving light verifications and benefit from a refined domain search of signatures.

2.3. SPECIFICATION-BASED MODELS

Ning and al. model

In [12], the authors described their on-going research on intrusion detection for mobile ad hoc networks. In particular, they employed specification-based techniques to monitor the ad hoc on-demand distance vector (AODV) routing protocol, a widely adopted ad hoc routing protocol. AODV is a reactive and stateless routing protocol that establishes routes only as desired by the source node. AODV is vulnerable to various kinds of attacks. The authors analyzed some of the vulnerabilities, specifically discussing attacks against AODV that manipulate the routing messages. They proposed a solution based on the specification-based intrusion detection technique to detect attacks on AODV. Their approach involves the use of finite state machines for specifying correct AODV routing behavior and distributed network monitors for detecting run-time violation of the specifications. In addition, one additional field in the protocol message is proposed to enable the monitoring. They illustrated that their algorithm, which employs a tree data structure and a node coloring scheme, can effectively detect most of the serious attacks in real time and with minimum overhead.

2.4. MODEL-BASED MODELS

Dyck model

Dyck model [13] is an example of static binary code analysis model-based intrusion detection.

It is the first efficient statically-constructed context-sensitive model. This model specifies both the correct sequences of system calls that a program can generate and the stack changes occurring at function call sites. Experiments demonstrate that the Dyck model is an order of magnitude more precise than a context-insensitive finite state machine model. With null call squelching, a dynamic technique to bound cost, the Dyck model operates in time similar to the context-insensitive model.

2.5. CRITICS

(i) The IDS based on expert systems may be a solution to a system intrusion problem, but it leads to some difficulties:

- The knowledge base of the expert system has to be always updated, which may lead to huge database.
- If the knowledge base is too large, then the inference engine might be too complex due to high number of rules to manage.

(ii) Neural networks (NNs) may also be a solution to such a problem, but they also present some difficulties:

- The behavior of a user may change from time to time, and some NNs can not deal with this. In this case, the NNs will fail to detect intruders.

(iii) Some existing languages for intrusion detection must be tested using realistic data in different operating systems: Linux, Solaris, or Windows NT, Sun, ... In addition to that, some known languages for intrusion detection are used only to detect known attacks (misuse detection). It will be interesting to study how these languages can be operable with an other language based on anomaly detection.

(iv) PCA may be a solution to intrusion detection problem. Even if, this method reduces the number of the original variables used, it leads to some difficulties:

- the exact value of the threshold (the distance) that determines if a command is an attack or not, is not given.
- we cannot estimite the dicrimination between the used variables. In fact, the discriminating power ratio is not used.

That is why our objective is to design an automatic tool based on the KM method, in order to increase the security audit trail analysis efficiency.

3. K-MEANS APPROACH

Our approach leads to find intruders in a given computer system by making an appropriate discrimination between the users clusters which define users profiles.

Of course, “Clustering” of personal behaviour is beset with basic difficulties which are problems of assessing numerical values to behaviour traits. That

is why and for simplifying the IDS problem; we considered in our study only one numerical criterion: frequency of use of UNIX commands.

Each user performs a set of commands (events) on a given system (Unix for example). The *events vector* of a given user profile is a binary vector, such that an element of this vector is equal to "1" if an event happens, and to "0" otherwise. These events may be:

- login;
- logout,
- read,
- execute,
- …

If a user u belongs to a users cluster C_i (having E_i as events vector) among k existing clusters, then its events vector E_i will be different from all the other events vector. Formally:

$\forall\ u \in$ (a cluster C_i) $\Rightarrow \forall\ j$=1..k, and $j \neq i$, then

$$E_i \neq E_j \qquad (1)$$

Equation [1] means that a user u belongs to one and only one users cluster. In addition to that, each users cluster C_i among k existing clusters consists at least of one user.

Formally :

$\forall$ (a users cluster C_i) such that i=1..k, then

$$\#C_i \geq 1 \qquad (2)$$

where :

$\#\ C_i$: the number of elements belonging to the cluster C_i.

Each users cluster C_k has a typical profile defined by the vector X_k, which is a weighted events vector E_k, such that each weight represents the number of occurrences of an event e_k. For example, a profile may be defined by the following characteristics:

- duration of a login session,
- CPU time consumed by a program,
- number of password failures during a minute,
- number of erased files in a login session,
- ...

Having k possible users clusters (C_1, C_2, ..., C_k), defined by their profile vectors, in a given organization (university, enterprise, …), if we find that a user u does not belong to its initial cluster C_i after applying KM algorithm, then we can say that this user is suspicious (an intruder).

Our approach to intrusion detection by using KM, follows three steps:

First step: auditing the system

Collect informations about all the system users, by auditing the system for a given period (12 months for example). These informations should include a number of profiles for each user, audited periodically (4 weeks a month, 5 days a week, and 8 hours a day) in an interval of time (during the login session, for example).

Then, define the *events vectors* E_u of each user u, and the *users profiles* X_u according to their usual actions or *behaviors*.

Second step: applying KM

Apply KM according to the collected informations and to the known groups of users.

Formulation of the KM problem

The KM [2] procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori. The main idea is to define k centroids, one for each cluster. These centroids shoud be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early groupage is done. At this point we need to re-calculate k new centroids as barycenters of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new centroid. A loop has been generated. As a result of this loop we may notice that the k centroids change their location step by step until no more changes are done. In other words centroids do not move any more. Finally, this algorithm aims at minimizing an *objective function*, in this case a squared error function. The objective function

$$J=\sum_{j=1..k} \sum_{i=1..n} \|x_i^{(j)} - c_j\|^2 \qquad (3)$$

Where $\|x_i^{(j)} - c_j\|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centre c_j, is an indicator of the distance of the n data points from their respective cluster centres.

The algorithm is composed of the following steps:

1. Place K points (**K users**) into the space represented by the objects that are being clustered. These points represent initial group centroids.
2. Assign each object (**user**) to the group that has the closest centroid.
3. When all objects (**users**) have been assigned, recalculate the positions of the K centroids.
4. Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects (**users**) into groups from which the metric to be minimized can be calculated.

Third step: testing and deducing the intruders

Each user will be assigned to the cluster having the shortest distance between its cluster centre and the user. If a user u does not belong to its initial cluster C_i after applying KM procedure, then this user will be suspicious (an intruder).

4. EXAMPLE

In a university, a network administrator extracted informations (audited files) from 1st January 2005 to 31th December 2005 (4 weeks a month, 5 days a week, and 8 hours a day), about twelve (12) students among 57. These informations concern the percentage of use of these commands:

($x1$) – *ftp*
($x2$) - *latex*
($x3$) – *telnet*
($x4$) – *rm*
($x5$) – *vi*
($x6$) – *chmod*
($x7$) – *xdvi*
($x8$) – *rlogin*
($x9$) – *lpr*
($x10$) – *finger*
($x11$) – *xlock*

In a same computer science class, one may find *bad* students, *medium* students, and *good* students. We assume here that a good Unix student is the one who uses more Unix commands than the other students. That is why, we suppose that the students must be separated into three clusters (k=3), according to a threshold (distance) which is the difference between their total average of the using rate of the previous commands during one year. The three clusters may determine for example three user classes:

- the *bad* students class (*having a mark less than 10/20*),
- the *medium* students class (*having a mark equal to 10/20*),
- and the *good* students class (*having a mark less than 15/20*).

The steps of applying KM are the following:

First step (see Table 1):

At the beginning, we assume that the users were first randomly assigned to clusters. Each user and his corresponding scores are detailed in Table 1 below. The three first clusters and their centres are:

- Cluster 1: 27 - 33 - 22 - 28 - 38 – 20 ; having cluster centre = 28
- Cluster 2: 51 - 45 - 44 – 40; having cluster centre = 45
- Cluster 3: 52 – 57 ; having cluster centre = 54.5

Second step (see Table 2):

After running the KM algorithm, the resulting new clusters and their centres (see Table 2) are:

- Cluster 1: 27 - 33 - 22 - 28 – 20; having cluster centre = 26
- Cluster 2: 45 - 44 - 40 – 38; having cluster centre = 41.75
- Cluster 3: 51 - 52 – 57; having cluster centre = 53.33

Third step (see Table 3):

At the end, the resulting clusters and their centres are the following (see Table 3):

- Cluster 1: 27 - 33 - 22 - 28 – 20; having cluster centre = 26
- Cluster 2: 45 - 44 - 40 – 38 ; having cluster centre = 41.75
- Cluster 3: 51 - 52 – 57; having cluster centre = 53.33

Table 1. First step of KM algorithm

Users / *Cluster centres*	27	51	52	33	45	22	28	44	40	38	20	57
User 27	0.00	0.65	0.68	0.16	0.49	0.14	0.03	0.46	0.35	0.30	0.19	0.81
User 51	0.65	0.00	0.03	0.49	0.16	0.78	0.62	0.19	0.30	0.35	0.84	0.16
User 52	0.68	0.03	0.00	0.51	0.19	0.81	0.65	0.22	0.32	0.38	0.86	0.14
Minimum	0.00	0.00	0.00	0.16	0.16	0.14	0.03	0.19	0.30	0.30	0.19	0.14
Cluster	1	2	3	1	2	1	1	2	2	1	1	3

Table 2. Second step of KM algorithm

Users / *Cluster centres*	27	51	52	33	45	22	28	44	40	38	20	57
28	0.03	0.62	0.65	0.14	0.46	0.16	0.00	0.43	0.32	0.27	0.22	0.78
45	0.49	0.16	0.19	0.32	0.00	0.62	0.46	0.03	0.14	0.19	0.68	0.32
54.5	0.74	0.09	0.07	0.58	0.26	0.88	0.72	0.28	0.39	0.45	0.93	0.07
Minimum	0.03	0.09	0.07	0.14	0.00	0.16	0.00	0.03	0.14	0.19	0.22	0.07
Cluster	1	3	3	1	2	1	1	2	2	2	1	3

Table 3. Last step of KM algorithm

Users / *Cluster centres*	27	51	52	33	45	22	28	44	40	38	20	57
26	0.03	0.62	0.65	0.14	0.46	0.16	0.00	0.43	0.32	0.27	0.22	0.78
41.75	0.49	0.16	0.19	0.32	0.00	0.62	0.46	0.03	0.14	0.19	0.68	0.32
53.33	0.74	0.09	0.07	0.58	0.26	0.88	0.72	0.28	0.39	0.45	0.93	0.07
Minimum	0.03	0.09	0.07	0.14	0.00	0.16	0.00	0.03	0.14	0.19	0.22	0.07
Cluster	1	3	3	1	2	1	1	2	2	2	1	3

At the beginning, *user 38* was in the cluster 1, after that, we find (him/her) in the cluster 2, (he/she) may be suspicious. *User 51* was in the cluster 2, and after that we find (him/her) in cluster 3, so (he/she) may be also suspicious.

5. EXPERIMENTS

In the same university, we achieve a serie of simulations where we consider 100 users assigned *randomly* to 3 clusters, according to their use of four commands: *ftp*, *latex, xdvi,* and *finger*. The following table is an example of collected data representing the occurences of the cited commands per studied user (Table 4).

We apply both KM and CDA methods to find suspicious users. The results of these simulations are the following (Table 5).

According to figure 1, KM is better than CDA in detecting intrusions. In fact, KM has a detection rate which is higher than the CDA one.

Table 4. Example of a data matrix

Commands / **Users (Observations)**	*ftp* (X1)	*latex* (X2)	*xdvi* (X3)	*finger* (X4)	**Cluster**
User1 (O1)	51	151	634	501	3
User2 (O2)	44	219	636	495	2
User3 (O3)	43	173	468	254	1
User4 (O4)	30	166	720	401	1
.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....
User100 (O100)	39	105	1259	258	2

Table 5. Comparison between K-means and CDA methods

IDS / Criteria	K-means	CDA
Anomaly score interval	**[0.55, 0.74]**	[0.41, 0.56]
Average detection rate	**64%**	47%

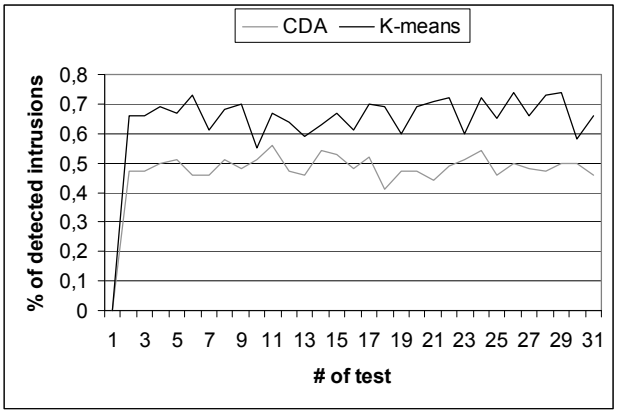


Fig. 1– Comparison between K-Means and CDA techniques.

6. CONCLUSION

Today, there are many IDSs, and each IDS has its advantages and its weaknesses. In the other hand, it is often difficult to compare IDSs because they do not use the same metrics (criteria). In this paper, we presented an anomaly detection method based on K-means. To test the robustness of our approach, we compared it to the CDA method while detecting anomalous profiles.

Our paper outlined the following ideas:

(i)- it introduced a survey of some intrusion detection methods,

(ii)- it presented our new approach based on the KM method, which includes the following steps :

- collecting informations (auditing the system),
- applying KM technique,
- testing and deducing the intruders.

(iii)- to illustrate the KM procedure, a simple example is given.

(iv)- to validate our approach, some experiments are presented.

Our experimental results show that our approach is better than CDA. In fact, our approach leads to an anomaly score interval that is higher than CDA one. In addition to that, the KM average detection rate is also higher than the CDA one. Nevertheless, our approach suffers from its high false alarm rate. Modelling behaviour traits remains a hard task, that is why we will focus soon on the enhancement of the KM method, in order to reduce the false alarm rate and to modelize other numerical criteria.

7. REFERENCES

[1] N. Ye, and X. Li, "A Scalable Clustering Technique for Intrusion Signature Recognition", from the *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security*, U.S Military Academy, West Point, NY, 5-6 June, pp. 1-4, 2001.

[2] J. B. MacQueen, "Some Methods for classification and Analysis of Multivariate Observations", *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability"*, Berkeley, University of California Press, 1:281-297, 1967.

[3] J. Marin, D. Ragsdale, and J. Surdu, "A Hybrid Approach to the Profile Creation and Intrusion Detection", technical report, Information Technology and Operations Center, United States Military Academy, 2000.

[4] A. Bivens, C. Palagiri, R. Smith, B. Szymanski, M. Embrechts, " Network-based Inrusion Detection Using Neural Networks", technical report, Rensselaer Polytechnic Institute, Troy, New York 12180-3590, 2002.

[5] R. Beghdad, "Canonical discriminant analysis for modelling and detecting intrusions in computer systems", *submitted for publication*.

[6] U. Lindqvist and P. A. Porras, "eXpert-BSM: A Host-based Intrusion Detection Solution for Sun Solaris", from *Proceedings of the 17th Annual Computer Security Applications Conference (ACSAC 2001)*, pp. 240–251, New Orleans, Louisiana, 2001.

[7] S. Cheung, U. Lindqvist and M. W. Fong, "Modeling Multistep Cyber Attacks for Scenario Recognition", from the Third *DARPA Information Survivability Conference and Exposition (DISCEX III)*, Volume I, pp. 284–292, Washington, D.C.2003.

[8] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang. "A novel anomaly detection scheme based on principal component classifier". In *Proceedings of the Third IEEE International Conference on Data Mining* (ICDM'03), pp. 172-179, Florida, Nov. 2003.

[9] B. Morin, H. Debar, "Correlation of Intrusion Symptoms : an Application of Chronicles", *In the Proceedings of the 6th Recent Advances in Intrusion Detection 2003 (RAID2003)*, 2003.

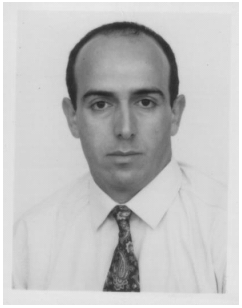
[10] K. Johansen and S. Lee, « CS424 Network Security: Bayesian Network Intrusion Detection (BNIDS), technical report, May 3, 2003.

[11] T. Abbes, A. Bouhoula, M. Rusinowitch, "Protocol Analysis in Intrusion Detection Using Decision Tree", *in the Proceedings of the International Conference on Information Technology Coding and Computing (ITCC'04)*, 2004.

[12] Peng Ning, Kun Sun, "How to Misuse AODV: A Case Study of Insider Attacks against Mobile Adhoc Routing Protocols,". In *Proceedings of the 4th Annual IEEE Information Assurance Worksho*p, pp. 60-67, West Point, June 2003.

[13] J. T. Giffin, S. Jha, B. P. Miller, "Efficient Context-Sensitive Intrusion Detection". In *11th*

Annual Network and Distributed Systems Security Symposium (NDSS), San Diego, California, February 2004.



Rachid BEGHDAD *received his computer science engineer degree in 1991 from the Polytechnical school of engineers, Algiers, Algeria. He received his Master computer science degree from Clermont-Ferrand University, France, in 1994. He earned his Ph.D. computer science degree from Toulouse University, France, in 1997.*
His main current interest is in the area of computer communication systems including intrusion detection methods, unicast and multicast routing protocols, real-time protocols, and wireless LAN protocols.