Improving the Quality of Alerts with Correlation in Intrusion Detection

Lalla Fatima SALIM [†] **and Abdellatif MEZRIOUI** [††]

[†] *FSTM, Mohammedia Morocco* [††] *INPT, Rabat Morocco*

Summary
With the growing deployment of networks and the Internet, the importance of network security has increased. Recently, however, systems that detect intrusions, which are important in security countermeasures, have been unable to provide proper analysis or an effective defense mechanism. Instead, they have overwhelmed human operators with a large volume of intrusion detection alerts. In this paper, we present an alert correlation technique based on causal relationships between alerts. The goal of the proposed technique is not only to group alerts together, but also to represent the correlated alerts in a way that they reflect the corresponding attack scenarios.

Keywords:
Intrusion alert, alerts correlation, attack scenarios, Network Security.

1. Introduction

With the development of network technologies and applications, network attacks are greatly increasing both in number and severity. As a key technique in network security domain, Intrusion Detection System (IDS) plays vital role of detecting various kinds of attacks and secures the network security and information infrastructures. The main purpose of IDS is to find out intrusions among normal audit data and this can be considered as classification problem.

Currently, there are two basic approaches to detection of an intrusion [1]. The first approach, called the anomaly detection (also called the behavioral detection), is to define and characterize the correct static form and the acceptable dynamic behavior of the system, and then to detect wrongful changes or wrongful behavior. The second approach, called the misuse detection (also called the signature detection), involves characterizing known ways to penetrate a system. Each known penetration method is usually described as a pattern. The misuse detection system looks for explicit patterns. The pattern may be a static bit string such as a specific virus bit string insertion. Alternatively, the pattern may describe a suspect set or sequence of actions.

Even though intrusion detection systems play an important role in protecting the network, they still have some weaknesses [2]. First, as network traffic increases, the intrusion detection alerts produced by IDSs are increasing exponentially. In spite of this increase, most IDSs neglect the overhead of human operators, who are overwhelmed by the large volume of alerts. Second, human operators are fully responsible for analyzing a network's status and the trends of cyber attacks. Third, although cyber attacks can produce multiple correlated alerts [3], IDSs are generally unable to detect such attacks as a complex single attack but regard each alert as a separate attack. Therefore, in the early stage, it is difficult to detect large-scale attacks such as a distributed denial of service (DDoS) or a worm.

These limitations are caused by the absence of a mechanism that can preprocess and correlate the massive number of alerts from IDSs. In fact, preprocessing and correlation of alerts are essential for human operators because the information reproduced by this means can reduce the overhead of human operators and help them react appropriately [4].

Research in the area of alert correlation has emerged in last few years and primarily concerns information modeling and high level reasoning. Current correlation approaches can be roughly divided into four categories: (1) similarity based approaches (e.g., [5, 6]), which perform clustering analysis through calculating the similarity between alert attributes, (2) approaches based on pre-defined attack scenarios (e.g., [2, 7]), which build attack scenarios through matching alerts to pre-defined scenario templates, (3) approaches based on prerequisites (pre-conditions) and consequences (post-conditions) of attacks (e.g., [8, 9]), which create attack scenarios through matching the consequence of one attack to the prerequisite of another, and (4) approaches based on multiple information sources (e.g., [10, 11, 12]), which correlate alerts from multiple security systems such as firewalls and IDSs.

Realizing the limitations of single detection mechanisms and systems, the alert correlation technique that we propose in this paper is aimed at reducing the alert overload by correlating results from multiple sensors to generate condensed views, reducing false positives by integrating network and host system information into the evaluation process and correlating events based on causal relationships to generate attack scenarios.

The remainder of this paper is organized as follows. In next section, we describe the architecture of our proposed system and the details of each component. In order to demonstrate the effectiveness of our solution, section 3 reports the experimental results. Finally we conclude and

Manuscript received December 5, 2007
Manuscript revised December 20, 2007

indicate some directions that could be followed in future work.

2. System architecture

As shown in Fig.1 our system consists of six components: the intrusion detection systems (sensors) that produce the alerts, the preprocessing function that converts alerts into a unified standard representation and eliminates false alerts, IDMEF alerts base where the alerts are stored, the clustering function that regroups alerts according different criteria, the Correlation function that attempts to discover causal relationships between alerts and knowledge base that contains information about monitoring hosts and networks. Each of these components is described in the following sections.

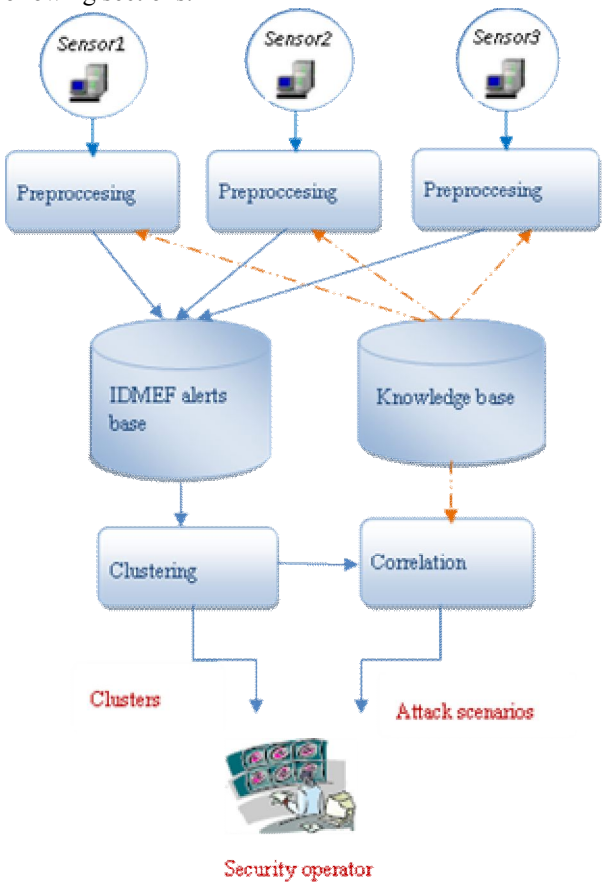

Fig.1 Overall system architecture.

2.1 Preprocessing function

In the recent years, researchers began to explore the benefits of collaboration among different IDS products. The main objective of the IDS cooperation is to reduce the number of alerts generated by correlating different IDS outputs and discard false alerts. By threading multiple alerts generated by related attacks, cooperating IDS modules will be able to provide a global view of intrusion activities.

In order to correlate alerts from multiple IDS products with different output formats, the preprocessing function convert the diversified formats into a unified standard representation. The format we chose is the Intrusion Detection Message Exchange Format (IDMEF) [13]. The purpose of the IDMEF is to define common data formats and exchange procedures for sharing information of interest to intrusion detection and response systems and those that may need to interact with them.

After converting alerts in the standard format, the preprocessing function delete the alerts that have one or more of their attributes values belonging to the set of invalid values. For example, an alert with an invalid time stamp must be deleted.

The third purpose of the preprocessing function is to eliminate false alarms based on information contained in the knowledge base. For example an alert generated in response to an attack that was exploiting a well-known vulnerability of a Windows 2000 operating system even though the system that the IDS was monitoring was a Linux operating system can be classified as a false alerts and can be deleted.

2.2 Knowledge base

At present, alert correlation techniques do not make full use of the information that is available. For example, they tend to only use the events generated by IDSs. We argue that alert correlation must take at least two information types into account: information related to the characteristics of the monitored information system and information about the vulnerabilities.

In our system, the knowledge base contains known vulnerabilities as well as the network and host asset information (e.g IP address, hostname, software installed in each host and their version ….).This information is compared with vulnerability requirement information to evaluate alerts and provide appropriate security solutions for real harmful attacks.

2.3 Clustering function

Alert clustering involves intelligently grouping or merging together identical alerts such that common generic attacks on systems are discovered [14].

In purpose to cluster alerts, we use the following set of characteristic features of sensor alerts:

− Attack-Type: This feature identifies the attack. Different sensors often refer to the same attacks with different attack names.

− Source-IP: This feature identifies where the attack is coming from. If the source IP address is not spoofed, it can identify the attacker.
− Target-IP: This feature identifies for whom the attack is meant.
− Source-Port: This feature identifies from where port the attack is coming.
− Target-Port: This feature identifies the port for which the attack is meant.
− Time: This feature denotes the time of the attack.
In this way an alert is denoted as:
A= {Attack-Type, Source-IP, Target-IP, Source-Port, Target-Port, Time}
In our approach the clustering function has a dual purpose:
The first purpose is to Group redundant alerts**:** When an attack occurs, the IDS may generate several alerts for this attack. The clustering function is used to recognize and group into an only one the alerts that actually correspond to the same occurrence of an attack.
To cluster and correlate alerts from heterogeneous sensors, we need to define the definite set of the attack class, and classify various attacks type into the corresponding class according to their characteristics.
This potential problem between heterogeneous sensors does not occur in our current developed system, since we project to use the same type of sensor. However, to enable our system to correlate heterogeneous sensors in an integrated system, we need to define the attack class.
In our current implementation with the same type of sensor, tow alerts A1 and A2 can be classified as redundant (and can be grouped on only one), if they have the same attack type, the same source and target IP address, the same source and target port and a ΔTime<=T.
When T is a delay after which two alerts will not be considered as redundant. Properly calibrating this delay is crucial to obtain good results. The experience demonstrates that this delay might depend on the attack type but for most attacks, this delay does not exceed two or three seconds.
Deleting the redundant alerts allows us to reduce the number of alerts transmitted to the correlation function. The experiment results demonstrate that we can reduce the number of alerts of 4% to 7%.
The second purpose of this component is to group the alerts into different clusters according to their source, target, time and attack-type. What allow the security operator to have an overall condensed view of the resources in the network: clustering on source attributes can help to associate alerts originating from the same sources. Clustering on time attributes can help to associate alerts that occur in short intervals. Clustering on attack types can help to associate alerts that are of the same nature.

2.4 Correlation function

The main objective of this component is to find the causal relationships between attacks (represented by alerts). We are interested in how individual attacks are combined to achieve the adversary's goal. The observation tells us that in a sequence of attacks, some attacks have to be performed earlier in order to launch later attacks. For example, an adversary always installs DDoS software before actually launching DDoS attacks. If we are able to capture these causal relationships, it may help us build stepwise of attack scenarios and reveal the adversary's attack strategy.
Attack scenarios are normally represented as attack graphs. These attack graphs can be manually constructed by security experts using knowledge such as topology and vulnerabilities of the protected network. But, this approach is time-consuming and error-borne.
In order to extract attack scenarios automatically, we introduce an alert causality Matrix that encode the weight that two alerts have a causal relationships between them.

	A1	A2	A3
A1	*W(A1,A1)*	*W(A1, A2)*	*W(A1, A3)*
A2	*W(A2, A1)*	*W(A2, A2)*	*W(A2, A3)*
A3	*W(A3, A1)*	*W(A3, A2)*	*W(A3, A3)*

Fig.2 Alert Causality Matrix

An Alert Causality Matrix for n alerts A1, A2, …, An is a not symmetric matrix with n × n cells, each of which contains a causality weight of two types of alert denoted W(ai, aj) . As shown in Figure 2 that represent an Alert Causality Matrix example of three alerts A1, A2 and A3, W(Ai, Aj) and W(Aj, Ai) represent two different causal relationships. W(Ai, Aj) suggests that alert Ai arrives before Aj and Aj is a consequence for Ai, while W(Aj, Ai) indicates that alert Aj arrives before Ai and Ai is a consequence of Aj. By distinguishing these two situations, one can gain better understanding of the causal relationship of these two types of attacks.
The value of each element in the causality matrix is between 0 and 1. It is computed as follows:
W (Ai, Aj) = Nc/N
N is the number of times that Aj arrive after Ai, and Nc is the number of times that Aj was a plausible consequence of Ai. In our approach an alert Aj is a plausible consequence of Ai if the following conditions hold:

Ai.time <= Aj.Time
AND {Ai.Target-IP = Aj.Target-IP
OR Ai.Target_IP = Aj.Source-IP}

After computing the weight causality for each two types of attack, the second step in our correlation engine is to construct the attack scenarios. For this purpose, we think that attack type, source and target IP address is the most significant attributes to discover the causal relationships between two alerts. The algorithm 1 shows our proposed method to correlate alerts in order to construct the attack scenarios.

Algorithm.1

```
    A: set of alerts, w: weight causality threshold
1.  For each alert Ai ∈ A do
2.    {
3.    For each alert Aj ∈ A with  Aj.Time <= Ai.Time
        do {
4.              If ( W(Aj,Ai)>=w )
              then {
5.                If ((Ai.Terget-IP = Aj.Target-IP) OR
                      (Aj.Terget-IP = Ai.Source-IP))
                  then
6.                    Connect Aj to Ai
                    }
              }
      }
7.      Draw attack scenarios
```

The following section explains the experimental results obtained by our correlation approach while using the 2000 DARPA intrusion detection scenario specific datasets (LLDOS 1.0 and LLDOS 2.0.2) [15].

3. Experiments on the DARPA DDoS evaluation dataset

To evaluate the effectiveness of our method in correlating alerts and its ability to construct attack scenarios, our first set of experiments was conducted in a simplified architecture with a single sensor. Our implementation of the method discussed in the last section is an off-line intrusion alert correlator. We use Java as the programming language, and Microsoft SQL Server 2000 as the DBMS to save the alert data set.

The experiment was conducted on the DARPA 2000 intrusion detection evaluation data-sets (LLDOS 1.0 and LLDOS 2.0.2) [15]. We used this dataset for which ground truth is known because it allowed us to assess the success of our experiments and compare our experimental results to work makes by other researchers in this area who have also used this dataset to report their results.

In the dataset, an intruder probes, breaks-in, installs the DDoS daemon, and launches a DDoS attack against an off-site server (Table 1). In our experiment, we use an alert log file [16] generated by RealSecure IDS. As a result of replaying the "Inside-tcpdump" file from DARPA 2000, Realsecure produces 922 alerts for LLDOS1.0 and 494 alerts for LLDOS2.0.2.

Table 1: Steps of the DDoS attack

Step	Attack
1	IPsweep for live host IPs from a remote site
2	Probe with SadmindPing the identified live IPs for sadmind daemon running on Solaris hosts
3	Compromise target hosts via the sadmind vulnerability (SadmindBOF)
4	Install the Trojan mstream DDoS software on compromised hosts
5	Register DDoS Trojan to the master computer
6	Launch the DDoS from compromised hosts against target

Figures 3 and 4 show the attack graphs (attack scenarios) extracted from the test data sets After applying the proposed correlation approach (for both LLDOS1.0 and LLDOS2.0.2), with causality threshold w = 0.25. The label inside each node is the attack type followed by the source IP address, target IP address and time of the node. Each edge denotes a causal relationship between the two end nodes.

The attack graph we extracted from LLDOS 1.0 (inside part) is partially shown in Figure 3. It correctly represents the DARPA DDoS attack scenario that is described in Table 1. The complete attack scenario can be divided into five stages. The first stage is missing in this attack scenario because RealSecure does not raise any alert for the ICMP probing activity executed by the attacker. Ning et al. report this same problem during their experiments with the DARPA data [9]. This highlights the fact that effectiveness of any high level analysis of sensor data is largely dependent on the quality of the sensor data itself.

The second stage consists of three Sadmind_Ping alerts, which the attacker used to find out the vulnerable Sadmind services. This intrusion alerts are from source IP address 202.077.162.213, and target IP addresses 172.016.112.010, 172.016.115.020, and 172.016.112.050, respectively. The third stage consists of some Sadmind_Amslverify_Overflow and Admind alerts. All these alerts have the same source IP address 202.077.162.213 and target IP addresses 172.016.112.10, 172.016.115.020 and 172.016.112.050, implying that the attacker tries several times to break into each victim running Sadmind service by using a buffer Overflow attack until one attempt succeeded. All the above three hosts were successfully broken into. The fourth stage consists of some Rsh alerts, with which the attacker installed and started the mstream daemon and master programs. In the same stage, Mstream_Zombie intrusion alerts corresponding to the communications between the mstream master and daemon programs is detected. Finally, the last stage consists of a DDOS alert. This alert has not been correlated into attack scenario because the attacker was using spoofed IP address. Instead, the Stream_DoS is

correlated with a Port_Scan alert as a separate attack scenario.

The experimental result For the LLDOS2.0.2 (inside part) is shown in figure 4. The attack scenario constructed for this attack has a similar pattern with the one extracted from LLDOS1.0, the attacker compromises two hosts 172.016.115.020 and 172.016.115.050 by exploiting the vulnerability of the Sadmind service, and installs DDoS daemons on these machines by using ftp. The first and five stages are also missed for the same reasons. In the same way the Stream_DoS is correlated with a Port_Scan alert as a separate attack scenario.

The correlation method to construct attack scenarios from intrusion alerts proposed by Ning and all [9] is similar to ours, the experimental results on the DARPA 2000 dataset show that both approaches provide the similar graph representation for attack scenarios. However, our approach is different than theirs in that it does not need to define a large number of rules in order to correlate alerts.

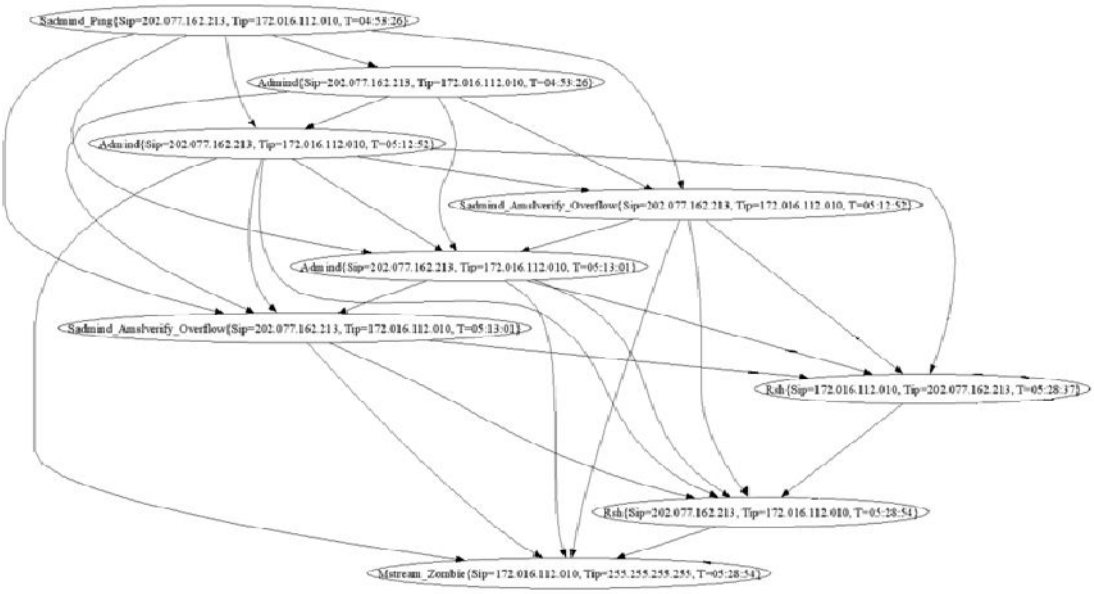

Fig.3 A part of Attack scenario for LLDOS1.0 (Attack scenario against 172.016.112.10).

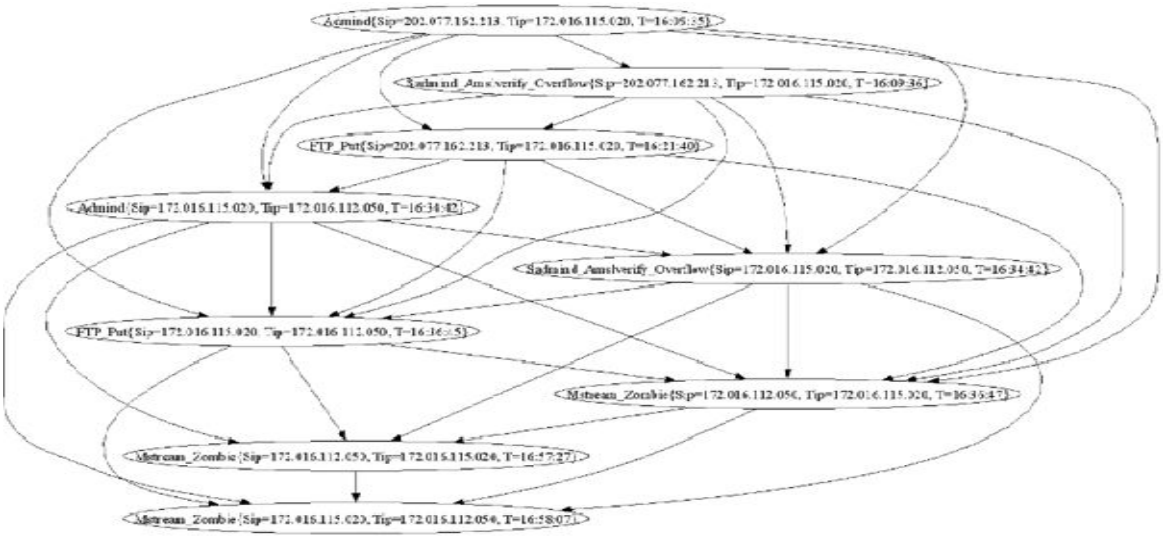

Fig.4 Attack scenario for LLDOS2.0.2.

4. Conclusion and Future work

In this paper, we have presented our alert correlation technique based on causal relationships between alerts. The goal of the proposed technique is not only to group alerts together, but also to represent the correlated alerts in a way that they reflect the corresponding attack scenarios. The experiments that we have performed using the DARPA 2000 intrusion detection scenarios specific datasets show that our technique can successfully correlate a large number of intrusion alerts and build stepwise of attack strategy.

This paper is a starting point for improving intrusion detection through alert correlation. In our future research, we plan to continue our investigation in this direction. In particular, we will develop additional techniques to reduce the number of alerts that is transmitted to correlation function, extend and use our alert correlation for the real-time and improve the correlation function in order to detect spoofed addresses.

References

[1] Jones AK, Sielken RS. Computer system intrusion detection: a survey. Technical report, Computer Science Department, University of Virginia; 2000.

[2] H. Debar and A. Wespi. Aggregation and correlation of intrusion-detection alerts. In Recent Advances in Intrusion Detection, LNCS 2212, pages 85 – 103, 2001.

[3] K. Kendall. A database of computer attacks for the evaluation of intrusion detection systems. Master's thesis. Massachusetts Institute of Technology; June 1999.

[4] E. Bloedorn, AD. Christiansen, W. Hill, C. Skorupka, LM. Talbot, J. Tivel. Data mining for network intrusion detection: how to get started. MITRE Technical Report; August 2001.

[5] A. Valdes and K. Skinner. Probabilistic alert correlation. In Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection (RAID 2001), pages 54–68, 2001.

[6] S. Staniford, J. Hoagland, and J. McAlerney. Practical automated detection of stealthy portscans. Journal of Computer Security, 10(1/2):105–136, 2002.

[7] B. Morin and H. Debar. Correlation of intrusion symptoms: an application of chronicles. In Proceedings of the 6th International Conference on Recent Advances in Intrusion Detection (RAID'03), September 2003.

[8] F. Cuppens and A. Miege. Alert correlation in a cooperative intrusion detection framework. In Proceedings of the 2002 IEEE Symposium on Security and Privacy, May 2002.

[9] P. Ning, Y. Cui, and D. S. Reeves. Constructing attack scenarios through correlation of intrusion alerts. In Proceedings of the 9th ACM Conference on Computer and Communications Security, pages 245–254, Washington, D.C., November 2002.

[10] P. Ning and D. Xu. Hypothesizing and reasoning about attacks missed by intrusion detection systems. ACM Transactions on Information and System Security, 7(4):591–627, November 2004.

[11] B. Morin, L. Mé, H. Debar, and M. Ducassé. M2D2: A formal data model for IDS alert correlation. In Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID 2002), pages 115–137, 2002.

[12] V. Yegneswaran, P. Barford, and S. Jha. Global intrusion detection in the domino overlay system. In Proceedings of the 11th Annual Network and Distributed System Security Symposium (NDSS'04), February 2004.

[13] D. Curry and H. Debar. "Intrusion Detection Message Exchange Format Data Model and ExtensibleMarkup Language (XML) Document Type Definition".draft-itetfidwg-idmef-xml-03.txt, February 2001.

[14] D. Yu and D. Frincke, "A Novel Framework for Alert Correlation and Understanding," Proceedings: International Conference on Applied Cryptography and Network Security (ACNS), Yellow Mountain, China, 2004.

[15] MIT Lincoln Laboratory, 2000 Darpa Intrusion Detection Scenario Specific Data Sets, 2000.

[16] North Carolina State University Cyber Defense Laboratory, Tiaa: A Toolkit for Intrusion Alert Analysis. http://discovery.csc.ncsu.edu/software/correlator/ver0.4/indx.html.

Lalla Fatima SALIM received the DESA degree in physical sciences and telecommunications from Faculty of Science of Marrakech, Morocco in 2001. She is currently pursuing the Ph.D. degree in Faculty of Science an technology of Mohammedia. Her research interests networks security.

Abdellatif MEZRIOUI received the PhD degree in the field of software process modeling from Med V University Rabat, Morocco in 2001. He is a professor at the INPT since 1995. His actual research domains are systems modeling, software engineering and networks security.