Modeling Network Traffic to Detect New Anomalies using Principal Component Analysis

Yacine Bouzida, Frédéric Cuppens, Sylvain Gombault

Modeling Network Traffic to Detect New Anomalies using Principal Component Analysis

Yacine Bouzida, Frédéric Cuppens and Sylvain Gombault
Département RSM GET/ENST Bretagne
02, rue de la Châtaigneraie CS 17607
F-35576 Cesson Sévigné
France
{*Yacine.Bouzida, Frederic.Cuppens, Sylvain.Gombault*}*@enst-bretagne.fr*

Abstract

We introduce a novel real time anomaly intrusion detection method using a multivariate statistical technique based on principal component analysis (PCA) to detect new anomalies. In fact, new attack forms are increasing each day and most of the current intrusion detection systems are signature based ones. As a result, these signature based tools fail to detect the new attacks. For this reason, network traffic modeling should be done in order to apply anomaly detection methods directly on the new modeled traffic. Different characteristics of the network traffic are analyzed, packet by packet, using PCA and significant statistical measures are considered to discover the difference between the normal (legitimate) and abnormal (called also illegitimate or attacks) traffic. An algorithm issued from the different statistical measures is discussed and the different results, performed over real time traffic corresponding to the different flooding DDoS attacks and the slammer worm that has infected more than $100,000$ vulnerable servers over internet in less than ten minutes, are presented.

Keywords

Principal Component Analysis, Statistical Measures, Anomaly Intrusion Detection, Novel Attacks.

1. Introduction

Intrusion detection is a challenging problem that has received much attention during the past two decades by many researchers. Most of the work done in this area is based on misuse detection. In fact, there are two main approaches; anomaly detection and misuse detection. The former consists in learning the normal behavior of a user (application, network traffic, or system events) profile and then observing the actual activities as reported in the audit data (network traffic, application data, etc.) to ultimately detect any significant deviations from the normal profiles. The latter consists in writing precise signatures and/or patterns in a database and then monitoring current system activities for such signatures or patterns and reporting the different matches.

With the increasing prolification of new high speed networks, intrusion detection systems should be improved to face the wide gap that is being created between the current IDSs and the emerging high speed networks. In our knowledge, there is not any tool being able to detect intrusions in the new generation networks since the majority of the methods used to discover the different signatures in the network traffic are NP Complete. As an

example Snort [**?**] used Boyer Moore technique then Aho Corasik/Boyer Moore from Silicon Defense [**?**] and Boyer-Moore-Horsepool from LANL/RADIANT [**?**] which are NP Complete rendering this tool unable to analyze traffic in high speed networks exceeding 30 Mbits/sec.

Current hackers know that the different intrusion detection tools deployed in our government, military and commercial computer systems are misuse based ones. This a priori knowledge of the different IDS tools encourages the attackers to implement new attacks that could not be detected by the deployed IDSs since they are most of the time signature based ones. The distributed Denial of Service (DDoS) attacks and the slammer [**?**] occurred respectively in 2000 and 2003 are such attack examples that were neither detected by the different IDSs nor prevented by the different access control tools or any other computer security mechanisms deployed for countering the devastating attacks. As a result of this situation, our computer systems are easily hacked and much confidential data are stolen and used maliciously by these new hackers.

Another problem of the actual IDSs is the time that should be invested by the system administrator to configure and manage the different deployed IDSs. Since the number of new vulnerabilities discovered each year is comprised between $1,000$ and $2,000$. A potential attacker has each day from 3 to 6 new vulnerabilities to exploit and the administrator must review continuously her security policy. However, many small companies lack dedicated computer security personnel and system administrators must play the role of the security site officer. Facing this dilemma, the attackers will never stop their malicious activities, against targeted companies or commercial sites, to reach their goals that may be political, economical, etc.

During the last decade a new intrusion detection model based on alert correlation that permits to reduce the amount of the alarms launched by the different sensors is widely investigated. There are many models and approaches that are introduced. CRIM [**?**] and CARDS [**?**] using a semi explicit correlation and others using an explicit model such as LAMBDA [**?**] and Adele [**?**]. Other implicit correlation models are widely investigated in the literature.

These correlation approaches aggregate and correlate alerts to reduce false positive alarms generated by the different sensors deployed in the audited system. Another task of these techniques is to discover attack scenarios when using explicit correlation and new attack scenarios in the case of semi explicit correlation.

Current correlation techniques use alerts from misuse intrusion detection tools. However, one main reproach to misuse detection is that they do not detect any new attack. In addition, more than 90% of the different alerts generated by these tools are false positives ones. The DARPA98 datasets [**?**] is an example where snort (configured with the recently updated signature database) generates thousands of alerts where 95% are false positives and only 5% are positive alarms corresponding to real attacks. Many known attacks can easily bypass all signature based detection tools. A simple method consists in modifying the freely downloadable source code of the different known attack tools.

Therefore, these correlation tools may not detect attack scenarios where all or some of its composing elementary attacks are not detected. In [**?**] virtual alerts are generated to fulfill the gaps in the ongoing scenario. This suggestion may not be anymore valuable since a lot of alerts are not detected when using these signature based IDS tools that generate much more false positives than positive alerts.

In this paper, we address the issue of improving the current IDSs in order to detect new attacks, consider their performance in high speed networks and take into account the administrator time constraint.

Our proposed method is based on principal component analysis (PCA) a multivariate statistical approach that has proven its efficiency in space reduction and is discussed in many application areas such image analysis, face recognition and time series prediction.

In [**?**], we used this method as a supervised technique to detect abnormal profiles. It consisted in modeling a user (application, system calls, ...) profile and then projecting all new profiles onto the new feature space generated by the new principal axes corresponding to the principal components. A decision is taken whether a new profile is normal or abnormal by only comparing its feature vector in the novel space generated by the principal components to the learned behaviors during the learning phase. In [**?**], we applied PCA as a reduction method before applying any learning machine algorithm to the KDD99 intrusion detection datasets [**?**]. The results outperformed much of the previous work done over these datasets. We demonstrated its efficiency in detecting high alarm rates in less time when compared to other methods.

In this paper, we use PCA not as a supervised technique but as an unsupervised one in which we do not have any knowledge about the attacks or normal traffic; i.e. the different datasets are not labeled a priori.

The rest of the paper is organized as the following. Section 2 presents related work in intrusion detection. Section 3 presents a description of the principal component analysis method then in section **??**, a network traffic modeling method using the Bro tool [**?**], in order to supply meaningful measures to the PCA engine, is presented. Finally, sections **??** and **??** respectively present the different results obtained from the experimentation and give some concluding remarks for future work.

2. Related Work

IDES [**?**] of SRI uses host based activities (e.g. CPU usage, file accesses, number of incorrect logins, etc.). The different normal activities are described as measures and their probability distributions are learned to be compared to new activities to detect any deviation. Forrest et al. [**?**] uses a technique that models the way an application or a service running on a machine normally behaves by registering the invoked sequences of system calls. An intrusion is assumed to exercise abnormal paths in the executable code, and is detected when new sequences are observed. Debar et al. [**?**] enhanced this method by considering fixed and variable length of the different system calls that are collected from a process execution. Their model minimizes the false negative alarm rate. In [**?**], Land et al. used a similar approach but they focused on an incremental algorithm that updates the stored sequences and uses data from UNIX shell commands. Schonlau et al. [**?**] monitored a set of UNIX users and collected a database of shell commands then discusses a statistical method based on principal regression model to discover masqueradors in the test datasets. Fan et al. [**?**] proposed a method that generates artificial anomalies to detect unknown and known network intrusions. They combined anomaly and misuse detection models that detect some known and unknown intrusions using the 10% of the KDD99 learning database [**?**].

Taylor et al. [**?**] introduced a lightweight approach anomaly approach that may be used with little human involvement in the system management and configuration. They implemented a statistical anomaly detection technique that may be deployed in high speed network for its speed and ease of use. However, a possible weakness of this method is a potential higher false positive rate and its design for a limited attack scope. They suggest to use only some IP header packets fields to analyze the traffic to detect the presence of attacks in the monitored network in real time. They used a cluster based method and

principal component analysis for data visualization in a $3D$ space. They suggested that a total population (packets) of between 250 to 350 yielded good cluster results [**?**].

Our proposal uses the different interesting ideas from the Taylor's work in identifying network intrusion by only using some fields of the IP packet header to discover new attacks. We use our knowledge and experience of principal component analysis in the field of intrusion detection to enhance the detection model of new attacks by investigating some statistical measures that can be easily used to detect those new attacks without a priori knowledge in real time.

3. Principal Component Analysis

The most common definition of PCA, due to Hotelling [**?**], is that, for a set of N observed $d - dimensional$ data vectors $v_i, i \in \{1, .., N\}$, the q principal axes $w_j, j \in \{1, .., q\}$, are those orthonormal axes onto which the retained variance under projection is maximal. It can be shown that the vectors w_j are given by the q dominant eigenvectors (i.e. those with the largest associated eigenvalues) of the simple covariance matrix $C = \sum_i \frac{(v_i - \overline{v})(v_i - \overline{v})^T}{N}$ such that $Cw_j = \lambda_j w_j$ and where $\overline{v}$ is the simple mean. The vector $u_i = W^T(v_i - \overline{v})$, where $W = \{w_1, w_2, \ldots, w_q\}$, is thus a $q - dimensional$ reduced representation of the observed vector v_i.

Therefore, principal component analysis (PCA) is a mathematical procedure that transforms a number of (possibly) correlated variables into a (smaller) number of uncorrelated variables called principal components. The objective of principal component analysis is to reduce the dimensionality (number of variables) of the dataset but retain most of the original variability in the data. The first principal component accounts for as much of the variability in the data as possible, and each succeeding component accounts for as much of the remaining variability as possible.

In this section we investigate the principal component analysis as a multivariate statistical approach. First, we examine the different principles of PCA before analysis and deciding whether data need to be standardized or not. Afterwards, we recall the definitions of some statistical measures that will be used in the next sections.

Before the calculation of the principal components, the data are mean-centered. This means that from each observation of a specific variable, the mean of all observations (of that variable) will be subtracted. This can be accomplished by making sure that the data set of the considered points is centered at the origin.

3.1 Data mean centering

Assume that we have N observed $d - dimensional$ data vectors $v_i, i \in \{1, .., N\}$. Therefore, the mean $\overline{v}$ may be calculated as the following:

$$\overline{v} = \frac{1}{N} \sum_{i=1}^{N} v_i \tag{1}$$

Then each new centered vector differs from the origin by:

$$\Phi_i = v_i - \overline{v} \tag{2}$$

A covariance matrix C is then considered to calculate the different Principal Compo-

nents (PCs):

$$C_{(d\times d)} = \frac{1}{N}\sum_{i=1}^{N} \Phi_i \Phi_i^T \tag{3}$$

The Principal components are then calculated as the eigenvectors of the covariance matrix $C_{(d\times d)}$.

In practice, it often occurs that different elements are completely different types of measurements. Some might be length, temperature, CPU time usage, etc. In such a case, the structure of the PCs will depend on the choice of units of measurement. Variable standardization is thus necessary.

3.2 Data standardization

The big drawback of PCA based on covariance matrices is the sensitivity of the PCs to the measurements units used for each element of the different variables. If there are large differences between the variances of elements of the different variables, then those variables whose variances are largest will tend to dominate the first few PCs. Therefore, we use a correlation matrix instead of a covariance matrix.

Having N observed $d - dimensional$ data vectors $v_i, i \in \{1,..,N\} = V_{(d\times N)}$, the correlation matrix R of the observed population is:

$$R_{(d\times d)} = \frac{1}{N}\sum_{i=1}^{N} \Upsilon_i \Upsilon_i^T = AA^T \tag{4}$$

where

$$\Upsilon_{i,j} = \frac{V_{ij} - \overline{v}_i}{\sigma_i}, \tag{5}$$

$$\sigma_i = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(V_{ij} - \overline{v}_i)^2} \tag{6}$$

and

$$A = \frac{1}{\sqrt{N}}[\Upsilon_1, \Upsilon_2, \ldots, \Upsilon_N] \tag{7}$$

σ_i corresponds to the standard deviation for each measurement of the population considered in the experiment.

In the case of standardized data, the principal components correspond to the eigenvectors of the correlation matrix R.

Another problem with the use of covariance matrices is that it is more difficult than with correlation matrices to compare informally the results from different analysis. Sizes of variances of PCs have the same implications for different correlation matrices of the same dimension, but not for different covariance matrices. Also, patterns of coefficients in PCs can be readily compared for different correlation matrices to see if the two correlation matrices are giving similar PCs, whereas informal comparisons are often much trickier for covariance matrices.

The loading coefficients are used in our experiments. They are the correlations between the original variables and the new variables. They are informative as they give an indication to which extent the original variables are important in forming new PCs and

then new variables. The higher the loading, the more influential the variable is in forming the principal components score and vice versa. Hence, if the loading coefficient between a new principal component and one variable from the dataset is positive then this indicates that the considered variable is very influential in forming that principal component.

We used the Pearson's correlation coefficient r which quantifies the measure of the strength of the association between two variables.

Considering two $n - dimensional$ variables X and Y, the formula for Pearson's correlation takes on many forms. A commonly used formula is expressed in equation **??**:

$$r = \frac{\sum_{i=1}^{n} X_i Y_i - \frac{1}{n} \sum_{i=1}^{n} X_i \sum_{i=1}^{n} Y_i}{\sqrt{\sum_{i=1}^{n} X_i^2 - \frac{1}{n} (\sum_{i=1}^{n} X_i)^2} \sqrt{\sum_{i=1}^{n} Y_i^2 - \frac{1}{n} (\sum_{i=1}^{n} Y_i)^2}} \tag{8}$$

Having these definitions and statistical measures on one hand, and facing the problem of intrusion detection and current high speed networks on the other hand, we use these statistical results to detect new network intrusions by checking only the different header packet fields and, for instance, a hash function value of the current analyzed packet payload. The following section presents the tool we used to generate the different interesting fields that are used by the principal component engine to detect new anomalies and a fast computational method to calculate the different eigenvectors and eigenvalues.

4. Network traffic modeling using Bro

There are many free IDSs tools available over Internet. Since we have a significant operational experience with Bro [**?**] and have analyzed this tool in a high speed network of up to 100 Mbps, we chose it as a traffic translator of real time network traffic into connection records that may be easily used by many machine learning and/or data mining algorithms. In fact, Bro is divided into an *event engine* that reduces a kernel-filtered network traffic stream into a series of higher-level events, and a *policy script interpreter* that interprets event handlers written in a proper language used to express a site's security policy. Furthermore, it comes with rich policy scripts that takes into account a variety of transport layer protocols. It also offers a powerful signature matching capability by using regular expressions to detect network intrusions.

We exploited the ability of this tool and we implemented a Bro event handler to output, in real time, a summarized record for each IP packet header.

Of course, we only use Bro as an event handler to output the different packets parsed fields in real time but not as an IDS as shown in figure **??**.

Since we are interested in using our tool in high speed networks, we only extract some features from the IP header of the packets to be analyzed and only, for the current experiments, a hash value of the packet's payload . Figure **??** shows these different features. Where :

1. **IP Source** coresponds to the source IP address of the packet (4 numbers for IPv4 and 6 for IPv6). In our experiments we are only interested in IPv4 protocol. The same feature vector will be used for the IPv6 protocol using 6 numbers.
2. **Port Src** is the IANA source port number of the packet.
3. **IP Dest** coresponds to the destination IP address of the packet (4 or 6 numbers).
4. **Port Dest** is the IANA destination port number of the packet.
5. **Protocol** corresponds to the protocol type of the packet; TCP, UDP, ICMP, etc.
6. **Packet Length** is the length of the packet.

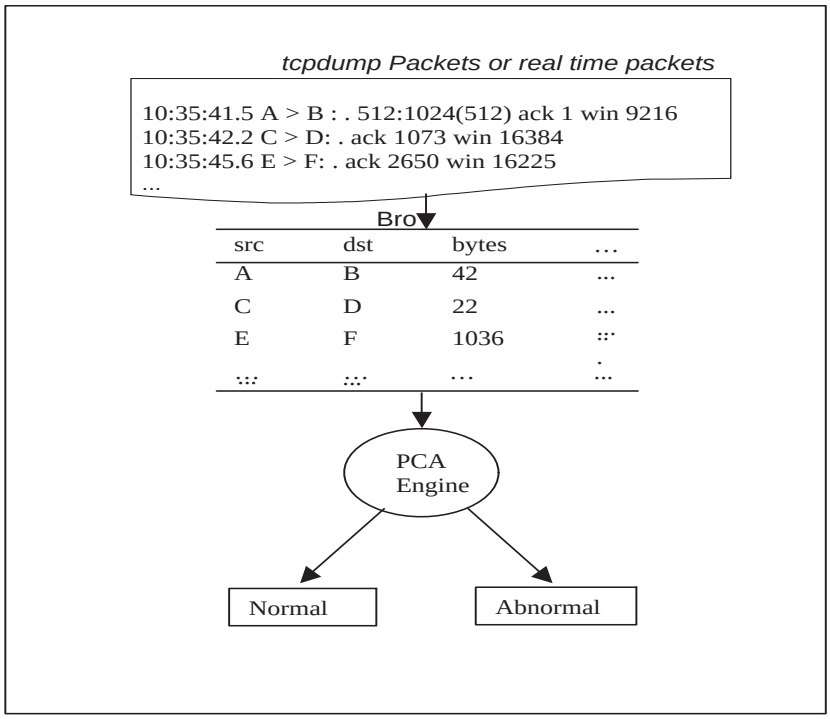

Figure 1: Bro and PCA Engine Architecture.

IP Source	Port Src	IP Dest	Port Dest	Protocol	Packet Length	Data Hash

Figure 2: The feature vector of a packet.

7. **Data Hash** corresponds to the signature value of the whole payload using an md5hash function.

The packets after being processed by the Bro tool as presented in **??** are then presented directly to the PCA Engine. We have already used PCA in learning users' profiles in a local area network [**?**], however, we used principal component as a supervised method where in the first stage, we learn the behaviors of the different users and then compare a new profile with the already learned profiles. In this paper, we use PCA as an unsupervised technique on the network traffic where we do not have any knowledge of the traffic and if whether it is normal or not. In addition, we use it here in real time and over network traffic. So, first we model the traffic and transform it into feature vectors as explained above and then apply PCA to the recently captured packets.

In our experiments we considered $d = 250$ to 350 packets to calculate the different statistical measures explained in the previous section; i.e:

1. **Standard deviation** of each principal component of the d considered packets,
2. **Proportion of variance** for each principal component, and
3. **Loading value**, using the Pearson's correlation coefficient (see equation **??**), for each feature with each principal component.

Since the different variables correspond to different statistical measures then a data standardization is necessary and a correlation matrix R should be calculated to find out the corresponding principal components. Each variable, in our experiments, is a d ($250 \leq d \leq 350$) dimensional vector, the matrix R is d by d and determining the d eigenvalues and eigenvectors is an intractable task when considering hundred of packets in real time and in the new generation high speed networks. So, a computational feasible method may be used to improve the processing time;

Let U_k be the k^{th} eigenvector of the correlation matrix $R_{d\times d}$ given in equation (4), λ_k the associated eigenvalue and
$\mathbf{U} = [\mathrm{U_1 U_2 \ldots U_d}]$ the matrix of these eigenvectors (that we call here eigenpackets). Then

$$RU_k = \lambda_k U_k \tag{9}$$

such that

$$U_k^T U_n = \begin{cases} 1 & \text{si } k = n \\ 0 & \text{si } k \neq n \end{cases} \tag{10}$$

Hence, from equations (**??**) and (4), we obtain

$$A\,A^T\,U_k = \lambda_k\,U_k \tag{11}$$

$$A^T A\,(A^T\,U_k) = \lambda_k\,(A^T\,U_k) \tag{12}$$

Let

$$Y_k = A^T\,U_k \tag{13}$$

then

$$A^T A\,Y_k = \lambda_k\,Y_k \tag{14}$$

From equation (**??**), Y_k is the eigenvector of $A^T A$ and λ_k is its corresponding eigenvalue.

Let $X_k = \alpha_k Y_k$, so X_k is also an eigenvector of $A^T A$
From equation (**??**)

$$X_k^T\,X_k = (\alpha_k\,A^T\,U_k)^T\,(\alpha_k\,A^T\,U_k) = \alpha_k^2 \lambda_k\,U_k^T\,U_k \tag{15}$$

we have $U_k^T\,U = 1$
then

$$X_k^T\,X_k = \alpha_k^2 \lambda_k \tag{16}$$

In order to obtain a normalized vector X_k ($i.e.\ X_k^T\,X_k = 1$) we shall have

$$\alpha_k^2 \lambda_k = 1 \Rightarrow \alpha_k = \frac{1}{\sqrt{\lambda_k}} \tag{17}$$

From equations (**??**) and (**??**), we have

$$A\,Y_k = A\,A^T\,U_k \tag{18}$$

$$A\,Y_k = \lambda_k\,U_k \tag{19}$$

$$U_k = \frac{1}{\lambda_k}\,A\,Y_k = \frac{1}{\lambda_k\,\alpha_k}A\,X_k = \frac{1}{\sqrt{\lambda_k}}A\,X_k \tag{20}$$

finally, we have

$$U_k = \frac{1}{\sqrt{\lambda_k}}A\,X_k \tag{21}$$

With this analysis, the calculation is highly reduced, from the order (of the number) of the considered packets in each batch ($250 \leq d \leq 350$) to the order of the number of the packet IP features considered for the analysis, because X_k is an eigenvector of the matrix $A^T A$ having a reduced dimension N ($N = 13$ features in our case).

This analysis is highly useful to use PCA in real time. However, [**?**] used the same method we introduced in [**?**] but did not manage to exploit this transformation in order to gain time and then their system, using MATLAB, consumes much more time than expected (more than 23 minutes instead of few seconds when using the above transformation).

5. Results and Discussions

To test our approach, we used a real network traffic generated from the different DDoS attacks tool such as Trinoo, Stacheldraht, TFN, etc. We are interested in the flooding traffic generated by the exploited agents of these tools over Internet to attack a victim. More precisely, we are interested in syn-flooding, smurfing. The second test concerns the probing attacks such as IPSweep and Portsweep using the well known nmap and superscan tools. And the last attack is the slammer worm that has infected thousands of machines over Internet in less than 10 minutes. For each of these attacks, we used the Bro tool to extract the different features and apply the different PCA steps as explained above. On the other hand we tested our method on a variety of normal traffic.

We present here just one dataset for each category because all the experiments we have done for each of them does not differ from one to the other in each sample for the same category.

In the following, we examine from table **??** the different statistical measures obtained from our analysis.

We mention that $Comp_i$ *Loading* corresponds to the Pearson's correlation coefficient between the more influential original variable and the i^{th} principal component. For example the smurfing traffic has a loading value of 0.99 concerning the first principal component in which the more influential variable in forming this first principal component is *IP Src*.

The PC_i *Loading* corresponds to the i^{th} principal component variance for the corresponding attack (or normal) traffic experiment.

All the results presented in table **??** are expected. In the case of the slammer attack, a vulnerable server launches many UDP/1434 packets to different addresses randomly calculated from the received packet whose goal is a buffer overflow exploit. These packets share the same characteristics as the received packet but are sent to different addresses over Internet to crash as many vulnerable servers as possible. In addition, the source IP address and the source port address remain unaltered as the packet length and the destination port (UDP/1434). However, the difference is in the destination IP address

Table 1 The different statistical measures obtained from the different experiments

Measures Traffic	PC1 Loading	PC2 Loading	PC1 Variance	PC2 Variance	cumulative variance
Normal	0.69(Port Src)	0.68(Port Dest)	0.71	0.28	0.99
IPSweep	0.98(IP Dest)	0.15(Port Dest)	0.49	0.48	0.97
Portsweep	0.98(Port Dest)	0.17(Port Src)	0.63	0.35	0.98
Smurfing	0.99(IP Src)	0.21(Data Hash)	0.42	0.20	0.62
synflood	0.96(Port Src)	0.95(IP Source)	0.83	0.16	0.99
Slammer	0.99(IP Dest)	0.11(Data Hash)	0.45	0.30	0.75

whose variance is very high (not showed in the table). Therefore the *IP Dest* feature is the variable that is much affecting the first principal component due to its substantial portion of the total variance in the original dataset captured from the network traffic concerning the slammer attack.

The smurf attack does not differ from the slammer attack according to our analysis since the same packet (echo reply) is from thousands machines against one victim. However, this attack traffic is collected in the victim side and the slammer's traffic is collected from the vulnerable machine (running a vulnerable mysql service) that has been attacked and that has started to attack thousands machines over Internet.

For IPsweep attack, the most influential variable for the first component is *IP Dest* feature since this attack consists in probing the different machines that are available in a target network. On the other hand, *Port Dest* is the most influential variable for the first principal component, concerning the Portsweep attack, since this attack tries to scan the different ports on a machine hoping to find out the different services that are available in that machine.

The syn-flooding attack consists in sending thousands of open connection packets against a server in order to overwhelm it and then renders its service unavailable. The tool we used exploits the IP spoofing technique and changes quickly its source ports numbers in order to be unnoticed by the target machine. This is why the first and the second PCs for this experiment are more affected by the *Port Src* and *IP Source* features.

We notice that the loading values of the first two principal components in the normal traffic are similar and the influential variables are *Port Src* and *Port Dest* since this situation corresponds to the balance in the variance of the different packets exchanged between a client and a server, and any other normal traffic between two machines, with respect to the source and destination ports.

Labib and Vemuri used in [**?**] some attacks from the DARPA98 [**?**] to validate their method. They propose a simple algorithm based on calculating a threshold T presented in equation (**??**)

$$T = \mid l_1 - l_2 \mid * p_v * 100 \tag{22}$$

where l_i corresponds to the loading value of the i^{th} principal component.

Therefore, they set a fixed value to which this threshold is compared. The value of

$C = 1$ is an example they give for their experiments to accommodate the various traffic patterns in the monitored network.

Since the data from the DARPA98 are not general and the different tools used to collect the different TCPDump traffic were not modified, they were used with their default form. Hence, an attacker changing the code of these tools, by modifying for example the Source IP address that may change in every packet launched against the victim, may never be detected with the above threshold presented in equation (**??**). As an example the syn-flooding attack (that corresponds to the syn flood attack of the TFN tool (available since 2000)) will never be detected since the loading of the first principal components are quite similar.

Using the different measures presented in table **??**, we propose a simple technique that is used in real time to detect the considered attacks:

BEGIN*Simple Algorithm to detect new attacks*
IF (the first two loadings are calculated using the source port and destination port) **THEN**
  IF ($0.60 \leq PC_i Loading_{i=1,2} \leq 0.80$)* **THEN**
    IF $T = \mid l_1 - l_2 \mid * p_v * 100 < 1$ **THEN**
      This is a normal traffic
    ELSE
      This is abnormal traffic (considered as a new attack that we do not know to what it corresponds, for the moment, therefore investigation should be done to discover to what this traffic corresponds to)
    ENDIF
  ENDIF
ELSE IF ($PC_1 Loading > 0.97$) **THEN**
    It is an attack corresponding to one of the above presented attacks.
  ELSE
    This is a new traffic that we consider for instance as abnormal.
    Investigation should be performed to discover to what this
    traffic corresponds to.
ENDIF
END

The above method is different from that presented in [**?**] since it does not only detect the different attacks as those experimented in this study but also detects those that are not considered in the experiments. However, for the different measures values that are not covered in the different attacks used in our experiment and that may appear in real life, we consider them as new attacks and investigation should be performed over the data traffic that has provided this new traffic whose measures are not met before. This last point is neither studied nor explained in [**?**]. This is the reason why we call this technique unsupervised approach since we do not have any ideas of the attacks we have not yet met.

Using the above simple algorithm with the different attacks considered we may obtain 100% of true detection rate and 0% of false alarms (false positive and false negative).

Since we do not have any knowledge of other new attacks, investigation should be done when new measures not covered by the above tests.

A Bi-Plot tool is used in [**?**] to visualize the different variables and network traffic in a

*The two boundaries 0.60 and 0.80 are issued from the different normal traffic datasets used in our different experiments not shown in table **??**.

$2D$ dimensional space. This is a good idea when we deal with an offline traffic analysis. However, in real life we cannot do so because in DoS (then DDoS) or Slammer attacks, we receive more than 60000 packets per second in our platform that does not permit to visualize the different measures in real time. However, the above algorithm is always valid since it is just generating alerts when it is necessary. On the other hand, a posterior analysis with PCA is always possible and a Bi-Plot tool may help the security officer to analyze the suspected traffic off line.

6. Conclusion

In this paper, we presented a new unsupervised anomaly detection algorithm using principal component analysis to detect new attacks. We investigated the principal component analysis and exploited the mathematical model that permits to reduce the space and calculate the eigenvectors and eigenvalues in a shorter time unlike the work of other researchers that have not taken into account the mathematical aspects of PCA in particular when we are confronted to the new generation high speed networks. Another advantage of the simplified algorithm that we have proposed, in addition to detecting the experimented attacks and differentiating them from normal network traffic, is its ability to detect new anomalies that are not considered in our test since the behavior of the network traffic is not finite and new protocols are investigated and added gradually to the Internet community. We believe that the robustness of our algorithm is not at maximum since we do not find new attacks that are different from those we have investigated in this paper. However, we should mention that other attacks of the type U2R (User to Root) and R2L (Remote to Local) [**?**] are not investigated in this paper. This is because we used only the packet header fields and did not investigate the payload. A hash value of the payload is used, as a summary of the payload, but it does not give high variance in the different experiments we conducted. We believe that investigating the payload in depth by considering new IP Packet fields in the packet feature vector may produce new attacks detection. This will be discussed in a forthcoming paper.

ACKNOWLEDGMENT

This work is financed in part by the ”Crédit Incitatif DDoS” from the GET (Groupe des Ecoles de Télécommunication). The authors thank all the members of this project; especially Hervé Debar and Olivier Paul.

References

[1] Y. Bouzida and S. Gombault. Intrusion Detection Using Principal Component Analysis. In *Proceedings of the 7th World Multiconference on Systemics, Cybernetics and Informatics*, Orlando, Florida, July 2003.

[2] Y. Bouzida and S. Gombault. Eigenconnections to Intrusion Detection. In *Security and Protection in Information Processing Systems*, Toulouse, France, August 2004. Kluwer Academic Publishers.

[3] J. McAlerney C. J. Coit, S. Staniford. Towards Faster String Matching for Intrusion Detection or Exceeding the Speed of Snort. Technical report, Silicon Defense, March 2001.

[4] F. Cuppens. Cooperative intrusion detection. In *International Sysmposium on Information superiority: tools for crisis and conflict-management*, Paris, France, September 2001.

[5] F. Cuppens and A. Miège. Alert Correlation in a Cooperative Intrusion Detection Framework. In *IEEE Symposium on Security and Privacy*, Oakland, USA, 2002.

[6] F. Cuppens and R. Ortalo. LAMBDA: A Language to Model a Database for Detection of Attacks. In *Third International Workshop on the Recent Advances in Intrusion Detection (RAID'2000)*, Toulouse, France, October 2000.

[7] DARPA Intrusion Detection Evaluation. Available at: http://www.ll.mit.edu/IST/ideval/data/data_index.html, 1998.

[8] H. Debar, M. Dacier, and M. A. Wespi. Fixed vs. Variable-Length Patterns for detecting Suspecious Process Behavior. In *Proceedings of the 5th European Symposium on Research in Computer Security*, pages 1–15, Louvain-la-Neuve, Belgium, September 1998.

[9] D. E Denning and P. G. Neumann. Requirements and model for IDES, a real time intrusion detection expert system. Technical report, Computer Science Laboratory, SRI International, 1985.

[10] W. Fan, M. Miller, S. J. Stolfo, W. Lee, and P. K. Chan. Using Artificial Anomalies to Detect Unknown and Known Network Intrusions. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 123–130, San Jose, CA, USA, November-December 2001.

[11] M. Fisk and G. Varghese. Applying Fast String Matching to Intrusion Detection. Technical Report CS2001-0670, Los Alamos National Laboratory, May 2001.

[12] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff. A Sense of Self for Unix Processes. In *IEEE Symposium on Security and Privacy*, Oakland, USA, 1996.

[13] H. Hotelling. Analysis of a complex statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

[14] KDD Cup 99 Intrusion Detection Datasets. Available at: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, 1999.

[15] K. Kendall. A Database of Computer Attacks for the Evaluation Of Intrusion Detection Systems, June 1999.

[16] K. Labib and V. R. Vemuri. Detecting and Visualizing Denial-of-Service and Network Probe Attacks Using Principal Component Analysis. In *Third Conference on Security and Network Architectures, (SAR'2004)*, La Londe, France, June 2004.

[17] T. Lane and C. Brodley. Approaches to on-line learning and concept drift for user identification in computer security. In *Proceedings of fourth International Conference on Knowledge Discovery and Data Mining (KDD-98)*, pages 259–263, 1998.

[18] C. Michel and Ludovic Mé. ADeLe: An Attack Description Language for Knowledge-Based Intrusion Detection. In *Trusted Information: The New Decade Challenge, IFIP TC11, Sixteenth Annual Working Conference on Information Security*, Paris, France, June 2001. Kluwer.

[19] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver. Inside the Slammer

Worm. *IEEE Security and Privacy*, 1(4), August 2003.

[20] M. Oka, Y. Oyama, H. Abe, and K. Kato. Anomaly Detection Using Layered Networks Based on Eigen Co-occurrence Matrix. In *Proceedings of the Seventh International Symposium on Recent Advances in Intrusion Detection (RAID'2004)*, pages 223–237, Nice, France, October 2004.

[21] V. Paxson. Bro: A system for detecting network intruders in real-time. *International Journal of Computer and Telecommunications Networking Archive*, 31(23-24), December 1999.

[22] M. Roesch. Snort - Lightweight Intrusion Detection for Networks. In *13th Systems Administration Conference - LISA 99*, 1999.

[23] M. Schonlau, W. DuMouchel, W. Ju, A. Karr, M. Theus, and Y. Vardi. Computer Intrusion: Detecting Masquerades. *Statistical Science*, 16(1):58–74, 2001.

[24] C. Taylor and J. Alves-Foss. An Empirical Analysis of NATE - Network Analusis of Anomalous Traffic Events.

[25] C. Taylor and J. Alves-Foss. NATE - Network Analusis of Anomalous Traffic Events, a low cost approach.

[26] J. Yang, P. Ning, X. S. Wang, and S. Jajodia. CARDS: A Distributed System for Detecting Coordinated Attacks. In *Proceedings of IFIP TC11 Sixteenth Annual Working Conference on Information Security*, pages 171–180, Washington DC, USA, August 2000.

Biography

Yacine Bouzida is a PhD student at GET/ENST Bretagne. He holds an engineering and a Master degree in computer science. His research fields focus on computer security. Currently, he is doing research in anomaly intrusion detection. He is also interested in alert correlation in intrusion detection, reaction and response mechanisms to intrusions, modeling network traffic for intrusion detection, network and information system topology for intrusion detection, and detection and prevention of distributed denial of service attacks. He is the first who has introduced and applied the Principal Component Analysis in Anomaly and then Misuse Intrusion Detection.

Frédéric Cuppens is a full professor at the GET/ENST Bretagne since 2003. He holds an engineering degree in computer science, a PhD and an HDR. He has been working for more than 15 years on various topics of computer security including definition of formal models of security policies, access control to network and information systems, intrusion detection and formal techniques to refine security policies and prove security properties. He has published more than 80 technical papers in refereed journals and conference proceedings. He is one of main contributors of the Or-BAC model and the designer of CRIM.

Sylvain Gombault is an assistant professor at the GET/ENST Bretagne. He holds an engineering degree in computer science from INSA Rennes. He first worked in industry and joined the ENST Bretagne in 1993. His research interests are related to networking security, intrusion detection (policy based and anomaly based) and reaction. He has been involved in several research projects on these topics. He is giving courses mainly on networking security architecture.