Collaborative Outlier Mining for Intrusion Detection

Goverdhan Singh*, Florent Masseglia*, Celine Fiot *, Alice Marascu *, Pascal Poncelet**

*INRIA Sophia Antipolis, 2004 route des lucioles - BP 93, 06902 Sophia Antipolis
Prenom.Nom@sophia.inria.fr
**LIRMM UMR CNRS 5506, 161 Rue Ada, 34392 Montpellier Cedex 5, France
poncelet@lirmm.fr

Résumé. Intrusion detection is an important topic dealing with security of information systems. Most successful Intrusion Detection Systems (IDS) rely on signature detection and need to update their signature as fast as new attacks are emerging. On the other hand, anomaly detection may be utilized for this purpose, but it suffers from a high number of false alarms. Actually, any behaviour which is significantly different from the usual ones will be considered as dangerous by an anomaly based IDS. Therefore, isolating true intrusions in a set of alarms is a very challenging task for anomaly based intrusion detection. In this paper, we consider to add a new feature to such isolated behaviours before they can be considered as malicious. This feature is based on their possible repetition from one information system to another. We propose a new outlier mining principle and validate it through a set of experiments.

1 Introduction

Protecting a system against new attacks, while keeping an automatic and adaptive framework is an important topic in this domain. One answer to that problem could rely on data mining. Actually, Data Mining for intrusion detection aims to provide new tools in order to detect cyber threats (Luo, 1999; Dokas et al., 2002; Bloedorn et al., 2001; Manganaris et al., 2000; Wu et Zhang, 2003). Among those data mining approaches, anomaly detection tries to deduce intrusions from atypical records (Lazarevic et al., 2003; Eskin et al., 2002). The overall principle is generally to build clusters, or classes, of usage and find outliers (*i.e.* events that do not belong to any class or group identifying normal usage). Actually, outlier detection aims to find records that deviate significantly from a well-defined notion of normality. It has a wide range of applications, such as fraud detection for credit card (Aleskerov et al., 1997), health care, cyber security (Bloedorn et al., 2001) or safety of critical systems (Fujimaki et al., 2005).

However, the main drawback of detecting intrusions by means of anomaly (outliers) detection is the high rate of false alarms since an alarm can be triggered because of a new kind of usages that has never been seen before (and is thus considered as abnormal). Considering the large amount of new usage patterns emerging in the Information Systems, even a weak percent of false positive will give a very large amount of spurious alarms that would be overwhelming for the analyst. Therefore, the goal of this paper is to propose an intrusion detection algorithm that is based on the analysis of usage data coming from multiple partners in order

to reduce the number of false alarms. Our main idea is that a new usage is likely to be related to the context of the information system on which it occurs (so it should only occur on this system). On the other hand, when a new security hole has been found on a system, the hackers will want to use it in as many information systems as possible. Thus a new anomaly that occurs on two (or more) information systems is probably not a new kind of usage, but rather an intrusion attempt. Let us consider A_x, an anomaly detected in the usage of web site S_1 corresponding to a php request on the staff directory for a new employee : John Doe, who works in room 204, floor 2, in the R&D department. The request will have the following form : `staff.php?FName=John\&LName=Doe \&room=204\&floor=2\&Dpt=RD`. This new request, due to the recent recruitment of John Due in this department, should not be considered as an attack. On the other hand, let us consider A_y, an anomaly that corresponds to a true intrusion. A_y will be based on a security hole of the system (for instance a php vulnerability) and might, for instance, look like : `staff.php?path=../etc/passwd%00`. One can see in this request that the parameters are not related to the data accessed by the php script, but rather to a security hole that has been discovered on the *staff* script. If two or more firms use the same script (say, a directory resquesting script bought to the same software company) then the usage of this security hole will certainly be repeated from one system to another and the request having parameter `../etc/passwd%00` will be the same for all the victims.

In this paper, we propose to provide the end-user with a method that takes only one parameter : n, the number of desired alarms. Then, based on the analysis of the usage data coming from the different partners, our algorithm will detect n common outliers they share. Such common outliers are likely to be true attacks and will trigger an alarm. In a real-world application of this technique, privacy preserving will be a major issue in order to protect partners' data. In this paper we focus on clustering and outlier detection techniques in a distributed environment. However, privacy issues in our framework are presented in another paper currently being submitted.

The paper is organized as follows. In Section 2 we present the motivation of this approach and our general framework and Section 3 gives an overview of existing works in this domain. Section 4 presents COD, our method for detecting outliers and triggering true alarms. Eventually, our method is tested through a set of experiments in Section 5 and Section 6 gives the conclusion.

2 Motivation and General Principle

Anomaly-based IDS (Eskin et al., 2002) can be divided into two categories ; semi-supervised and unsupervised. The semi-supervised methods build a model of "normal" behaviours on the system. Every behaviour that is not considered as normal is an anomaly and should trigger an alarm. The unsupervised methods do not use any labelled data. Usually, based on a clustering algorithm, they try to detect outliers and consider them as anomalies. Obviously, anomaly-based IDS will suffer from a very high number of false alarms since a new kind of behaviour will be considered as an anomaly (and an attack). In this paper, we propose to improve the results of unsupervised IDS by means of a collaborative framework involving different network-based systems. Section 3 gives an overview of the existing IDS based on the principles presented above and the existing collaborative IDS. However, to the best of our knowledge, our proposal is the first unsupervised IDS using the common anomalies of multiple partners in or-

der to detect the true intrusion attempts. The main idea of our proposal is that multiple partners do not share the same data, but they share the same systems (the Web server can be Apache or IIS, the data server can run Oracle, the scripts accessing the data can be written with PHP or CGI, etc). When a security hole has been found for one system (say a php scripts with specific parameters leading to privileged access to the hard drive), then this weakness will be the same for all the partners using the same technology. Our goal is to reduce the rate of false alarms based on this observation, as explained in section 2

In this paper we present COD (Common Outlier Detection) a framework and algorithm intended to detect the outliers shared by at least two partners in a collaborative IDS. Outliers are usually small clusters. Some methods used to find them are presented in section 3. Our goal is to use such outlier lists from different systems (based on a similar clustering, involving the same similarity measure). If an outlier occurs for at least two systems, then it is considered as an attack. COD is indeed based on the assumption that an intrusion attempt trying to find a weakness of a script will look similar for all the victims of this attack. We propose to detect intrusion attempts among the records of a Web server, such as an Apache access log file. Such a file keeps record, for each access on the Web site, of the IP, date, requested URL and referrer (among other informations). For clarity of presentation we present our framework on the collaboration of two Web sites, S_1 and S_2 and we consider the requests that have been received by the scripts of each site (cgi, php, sql, etc). Our goal is to perform a clustering on the usage patterns of each site and find the common outliers. However, that would not be enough to meet the second constraint of our objective : to require only one parameter, n, the number of alarms to return. Our similarity measure (presented in section 4) will allow normal usage patterns to be grouped together rather than grouped with intrusion patterns. On the other hand, our similarity measure also has to ensure distinguishing an intrusion pattern from normal usage patterns and from other intrusion patterns (since different intrusion patterns will be based on a different security hole and will have very different characteristics). Our algorithm performs successive clustering steps for each site. At each step we check the potentially matching outliers between both sites. The clustering algorithm is agglomerative and depends on the maximum dissimilarity (MD) that has to be respected between two objects.

This work is intended to explore the solutions for monitoring a network in real time. Then, the potential alarms will be triggered at each step of the monitoring (for instance with a frequency of one hour). Depending on the number of true or false alarms, the user might want to adjust n for the next step, until no (or very few) false alarm is returned. Our assumption is that common outliers, sorted by similarity from one site to another, will give the intrusions at the beginning of the list. Our challenge, in this paper, is to reply to important questions underlying our method ; what is the similarity between two usage patterns ? How to separate clusters in order to give the list of outliers ? How to detect common outliers ? Our main algorithm, corresponding to the framework presented in this section, is given in section 4.

3 Related Works

Over time many techniques have been developped to detect outliers, leading to a number of surveys and review articles (Hodge et Austin, 2004; Chandola et al., 2008). Some of them more precisely focus on the topic of outlier detection within the context of intrusion detection in computer networks (Lazarevic et al., 2003; Patcha et Park, 2007). In this paper, we focus

on this specific area and we propose an *unsupervised* anomaly-based detection system. On the opposite to *semi-supervised* anomaly detection systems, consisting of describing normal behaviours to detect deviating patterns (Marchette, 1999; Wu et Zhang, 2003; Vinueza et Grudic, 2004), *unsupervised* techniques do not require a preliminary identification of the normal usage by a human expert. Our application will thus be more usable in a real-world context.

Statistic community has quite extensively studied the concept of outlyingness (Barnett et T. Lewis, 1994; Schölkopf et al., 2001; Markou et Singh, 2003; Kwitt et Hofmann, 2007). Statistical approaches construct probability distribution models under which outliers are objects of low probability (Rousseeuw et Leroy, 1996; Billor et al., 2000; Lee et Xiang, 2001) However, within the context of intrusion detection, dimensionality of data is high. Therefore, to improve overall performance and accuracy, it has become necessary to develop data mining algorithms using the whole data distribution as well as most of data features (Knorr et Ng, 1998; Breunig et al., 2000; Aggarwal et Yu, 2001).

Most of these approaches are based on clustering-based outlier detection algorithms (Jain et Dubes, 1988; Ng et Han, 1994; Ester et al., 1996; Portnoy et al., 2001; Tax et Duin, 2001; Eskin et al., 2002; He et al., 2003; Papadimitriou et al., 2003). Such techniques rely on the assumption (Chandola et al., 2008) that normal points belong to large and dense clusters while anomalies (or outliers, atypical instances) either do not belong to any clusters (Knorr et Ng, 1998; Ramaswamy et al., 2000; Duan et al., 2006) or form very small (or very sparse) clusters (Otey et al., 2003; Chimphlee et al., 2005; Pires et Santos-Pereira, 2005; Fan et al., 2006; Ceglar et al., 2007). In other words anomaly detection consists in identifying those among the data that are far from significant clusters – either isolated or in small clusters.

On the contrary, misuse techniques (*i.e.* approaches that detect elements similar to well-known malicious usage) will precisely detect attacks but they will miss every intrusion that differs from these already known attack signatures. Therefore some works proposed collaborative frameworks in order to improve performance and both true and false alarm rates (Valdes et Skinner, 2001; Yegneswaran et al., 2004). These approaches rely on propagating in a distributed IDS IP blacklist after individual misuse or anomaly detection. Also this communication can lead to more accurate results, it does not allow the system to uncover totally unknown attacks or to avoid high false alarm rates. For these reasons we propose in this paper an anomaly detection approach that uses collaboration between systems in order to discriminate attacks from emerging or novel usage behaviours, thus leading to a reduced number of false alarms.

Depending on the approach, the number of parameters required to run the algorithm can be high and will lead to different outliers. To avoid this, some works return a ranked list of potential outliers and limit the number of parameters to be specified (Ramaswamy et al., 2000; Jin et al., 2001; Fan et al., 2006).

4 COD : Common Outlier Detection

The principle of COD is to perform successive clustering steps on usage patterns of different partners sites, until the number of common outliers meets the number of alarms desired by the user. We present in this section an algorithm designed for two information systems. Extending this work to more than two systems would require a central node coordinating the comparisons and triggering the alarms, or a peer-to-peer communication protocol. This is not the goal of this paper. Our objects are the parameters given to script files in the requests re-

ceived on a Web site. In other words, the access log file is filtered and we only keep lines corresponding to requests with parameters to a script. For each such line, we separate the parameters and for each parameter we create an object. Let us consider, for instance, the following request : `staff.php?FName=John&LName=Doe`. The corresponding objects are o_1 =`John` and o_2 =`Doe`. Once the objects are obtained from the usage data of multiple Web sites, COD is applied and gives their common outliers.

Algorithm Cod
Input : U_1 and U_2 the usage patterns of sites S_1 and S_2 and n the number of alarms.
Output : I the set of clusters corresponding to malicious patterns.

1. Build M, the distance matrix between each pattern ;
2. $\forall p \in M, Neighbours_p \leftarrow$ sorted list of neighbours for p (the first usage pattern in the list of p is the closest to p).
3. $DensityList \leftarrow$ sorted list of patterns by density ;
4. $MD \leftarrow 0$;
5. $MD \leftarrow MD + 0.05$;
6. $C_1 \leftarrow Clustering(U_1, MD)$;
   $C_2 \leftarrow Clustering(U_2, MD)$;
7. $O_1 \leftarrow Outliers(C_1)$; $O_2 \leftarrow Outliers(C_2)$;
8. $I \leftarrow CommonOutliers(O_1, O_2, MD)$;
9. If $|I| \leq n$ then return I ;
10. If $MD = 1$ then return I ; // No common outlier
11. Else return to step 5 ;

End algorithm Cod

As explained in section 2, COD algorithm will process the usage patterns of both sites step by step. For each step, a clustering result is provided and analyzed for intrusion detection. First, MD is set to obtain very tight and numerous clusters (very short similarity is allowed between two objects in a cluster). Then, MD is relaxed by an amount of 0.05 step after step in order to increase the size of resulting clusters, decrease their number and lower the number of alarms. When the number of alarms desired by the user is reached, then COD ends.

4.1 Clustering

Algorithm Clustering
Input : U, the usage patterns
and MD, the Maximum Dissimilarity.
Output : C, the set of as large clusters as possible,
respecting MD.

1. $i \leftarrow 0$; $C \leftarrow \emptyset$;
2. $p \leftarrow$ next unclassified pattern in $DensityList$;
3. $i++$; $c_i \leftarrow p$;
4. $C \leftarrow C + c_i$;

5. $q \leftarrow$ next unclassified pattern in $Neighbours_p$;
6. $\forall o \in c_i$
   If $d(o, q) > MD$ then return to step 2 ;
7. add q to c_i ;
8. $C_c \leftarrow LCS(C_c, q)$; //C_c is the center of C
9. return to step 5 ;
10. If unclassified patterns remain then return to step 2 ;
11. return C ;

End algorithm Clustering

COD Clustering algorithm is based on an agglomerative principle. The goal is to increase the volume of clusters by adding candidate objects, until the Maximum Dissimilarity (MD) is broken (*i.e.* there is one object o_i in the cluster such that the similarity between o_i and the candidate object o_c is greater than MD).

Similarity between objects. We consider each object as a sequence of characters. Our similarity is then based on the longest common subsequence (LCS), as described in definition 1.

Definition 1 *Let s_1 and s_2 be two sequences. Let $LCS(s_1, s_2)$ be the length of the longest common subsequences between s_1 and s_2. The* dissimilarity *$d(s_1, s_2)$ between s_1 and s_2 is defined as follows : $d(s_1, s_2) = 1 - \frac{2 \times LCS(s_1, s_2)}{|s_1|+|s_2|}$*

Example 1 *Let us consider two parameters p_1=`intrusion` and p_2=`induction`. The LCS between p_1 and p_2 is L=`inuion`. L has length 6 and the similarity between p_1 and p_2 is $d = 1 - \frac{2 \times L}{|p_1|+|p_2|} = 33.33\%$. Which also means a similarity of 77.77% between both parameters.*

Centre of clusters. When an object is inserted into a cluster we maintain the centre of this cluster, since it will be used in the CommonOutliers algorithm. The centre of a cluster C is the LCS between all the objects in C. When object o_i is added to C, its center C_c is updated. The new value of C_c is the LCS between the current value of C_c and o_i.

4.2 Wavelet-based Outlier Detection

Most previous work in outlier detection require a parameter (Zhong et al., 2007; Portnoy et al., 2001; Joshua Oldmeadow et al., 2004), such as a percent of small clusters that should be considered as outliers, or the top-n outliers. Their key idea is generally to sort the clusters by size and/or tightness. We consider that our clusters will be as tight as possible, according to our clustering algorithm and we want to extract outliers by sorting the cluster by size. The problem is to separate "big" and "small" clusters. Our solution is based on an analysis of cluster distribution, once they are sorted by size. The usual distribution of clusters is illustrated by Figure 1 (screenshot made with our real data). We propose to use a wavelet transform to cut down the distribution. In figure 1, the y axis stands for the size of the clusters, whereas their index in the sorted list is represented on x, and the two plateaux allow separating small and big clusters.

The wavelet transform is a tool that cuts up data or functions or operators into different frequency components, and then studies each component with a resolution matched to its scale

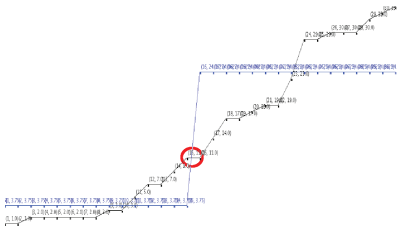

FIG. 1 – *Detection of outliers by means of Haar Wavelets*

(Daubechies, 1992). Mathematically, the continuous wavelet transform is defined by :
$T^{wav} f(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} f(x)\psi^*(\frac{x-b}{a})dx$

where z^* denotes the complex conjugate of z, $\psi^*(x)$ is the analyzing wavelet, a (> 0) is the scale parameter and b is the translation parameter. We use the Haar wavelets to illustrate our outlier detection method. Let us consider the following series of values : $[1, 1, 1, 2, 7, 10, 11, 12]$. Its Haar wavelet transform is illustrated by the following table :

Level	Approximations	Coefficients
8	1, 1, 1, 2, 7, 10, 11, 12	
4	1, 1.5, 8.5, 11.5	0, -0.5, -1.5, -0.5
2	1.25, 10	-0.25, -1.5
1	5.625	-4.375

Then, we keep only the most two significant coefficients and we make the others zero. In our series of coefficients ($[5.625, -4, 375, -0.25, -1.5, 0, -0.5, -1.5, -0.5]$) the most two significant ones are 5.625 and $-4, 375$, meaning that the series becomes $[5.625, -4, 375, 0, 0, 0, 0, 0, 0]$. In the following step, the inverse operation is calculated and we obtain an approximation of the original data $[1.25, 1.25, 1.25, 1.25, 10.0, 10.0, 10.0, 10.0]$. This gives us two plateaux and allows cutting the series after index 4 in order to separate big and small values.

4.3 Comparing Outliers

Since we want our global algorithm to require only one parameter (the number of alarms), we want to avoid introducing a similarity degree for comparing two lists of outliers. For this comparison, CommonOutliers algorithm will use the centre of outliers. For each pair of outliers, CommonOutliers calculates the similarity between centers of these outliers. If this similarity is below the current MD (C.f. Subsection 4.1), then we consider those outliers as similar and add them to the alarm list.

Algorithm CommonOutliers
Input : O_1 and O_2, two lists of outliers and MD, the maximum dissimilarity.
Output : A, the list of alarms (common outliers).

1. $A \leftarrow \emptyset$
2. $\forall i \in O_1$ do

3. $\forall j \in O_j$ do
4. $centre_i \leftarrow centre(i)$;
5. $centre_j \leftarrow centre(j)$;
6. If $d(centre_i, centre_j) < MD$
   Then $A \leftarrow A + i \cup j$;
7. done ;
8. done ;
9. Return A ;

End algorithm CommonOutliers

5 Experiments

The goal of this section is to analyze our results (*i.e.* the number of outliers and true intrusions and the kind of intrusions we have detected). Our datasets come from two different research organizations ; (*anonymized for submission*). We have analyzed their Web access log files from March 1 to March 31. The first log file represents 1.8 Gb of rough data. In this file, the total number of objects (parameters given to scripts) is 30,454. The second log file represents 1.2 Gb of rough data and the total number of objects is 72,381. COD has been written in Java and C++ on a PC (2.33GHz i686) running Linux with 4Gb of main memory. Parameters that are automatically generated by the scripts have been removed from the datasets since they cannot correspond to attacks (for instance "`publications.php?Category=Books`"). This can be done by listing all the possible generation of parameters in the scripts of a Web site.

As described in Section 2, COD proceeds by steps and slowly increases the value of MD, which stands for a tolerance value when grouping objects during the clustering process. In our experiments, MD has been increased by steps of 0.05 from 0.05 to 0.5. For each step, we report our measures in table 1. The meaning of each measure is as follows. O_1 (resp. O_2) is the number of outlying objects in site 1 (resp. site 2). $\%_1$ (resp $\%_2$) is the fraction of outlying objects on the number of objects in site 1 (resp. site 2). For instance, when MD is set to 0.3, for site 1 we have 5,607 outlying objects, which represents 18.4% of the total number of objects (*i.e.* 30,454) in site 1. COD is the number of common outliers between both sites and $\%_{FA}$ is the percentage of false alarms within the common outliers. For instance, when MD is set to 0.05, we find 101 alarms among which 5 are false (which represents 4.9%). One first observation is that outliers cannot be directly used to trigger alarms. Obviously, a number as high as 5,607 alarms to check, even for one month, is not realistic. On the other hand, the results of COD show its ability to separate malicious behaviour from normal usage. Our false alarms correspond to normal requests that are common to both sites but rarely occur. For instance, on the references interrogation script of *anonym_lab1*, a user might request papers of "John Doe" and the request will be
`publications.php?FName=John\&LName=Doe`. If another user requests papers of "John Rare" on the Web site of *anonym_lab2*), the request will be
`biblio.php?FName=John\&LName=Rare` and the parameter "John" will be given as a common outlier and trigger an alarm. As we can see, $\%_{FA}$ is very low (usually we have at most 5 false alarms in our experiments for both Web sites) compared to the thousands of outliers that

	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
O_1	13197	10860	8839	7714	6547	5607	5184	4410	3945	3532
$\%_1$	43.3%	35.6%	29%	25.3%	21.5%	18.4%	17%	14.4%	12.9%	11.6%
O_2	35983	27519	24032	20948	18152	14664	12738	11680	10179	8734
$\%_2$	49.6%	37.9%	33.1%	28.9%	25%	20.2%	17.5%	16.1%	14%	12.1%
COD	101	78	74	70	67	71	71	85	89	90
$\%_{FA}$	4.9%	5.12%	4%	2.85%	1.5%	2.8%	2.8%	10.6%	11.2%	16.6%

TAB. 1 – *Results on real data*

have been filtered by COD. Another lesson from these experiments is that a low MD implies very small clusters and numerous outliers. These outliers are shared between both sites, among which some are false alarms due to rare but common normal usage. When MD increases, the clustering process gets more agglomerative and alarms are grouped together. Then one alarm can cover several ones of the same kind (*e.g.* the case of easter eggs explained further). At the same time, the number of outliers corresponding to normal usage decreases (since they are also grouped together). Eventually, a too large value of MD implies building clusters that do not really make sense. In this case, outliers will get larger, and the matching criteria will get too tolerant, leading to a large number of matching outliers capturing normal usage. In a streaming environment involving the real data of these experiments, one could decide to keep 70 as the number of desired alarms and watch the ratio of false alarms. If this ratio decreases, then the end-user should consider increasing the number of desired alarms.

6 Conclusion

In this paper, we have proposed i) an unsupervised clustering scheme for isolating atypical behaviours, ii) a parameterless outlier detection method based on wavelets and iii) a new feature for characterizing intrusions. This new feature is based on the repetition of an intrusion attempt from one system to another. Actually, our experiments show that atypical behaviours cannot be directly used to trigger alarms since most of them correspond to normal requests. On the other hand, this very large number of outliers can be effectively filtered (reducing the amount of atypical behaviours up to 0.21%) in order to find true intrusion attempts (or attacks) if we consider more than one site. Eventually, our method guarantees a very low ratio of false alarms, thus making unsupervised clustering for intrusion detection effective, realistic and feasible.

Références

Aggarwal, C. C. et P. S. Yu (2001). Outlier detection for high dimensional data. *SIGMOD Records 30*(2), 37–46.

Aleskerov, E., B. Freisleben, et B. Rao (1997). Cardwatch : A neural network based database mining system for credit card fraud detection. In *IEEE CIFE*.

Barnett, V. et T. T. Lewis (1994). *Outliers in statistical data*. John Wiley & Sons.

Billor, N., A. S. Hadi, et P. F. Velleman (2000). BACON : blocked adaptive computationally efficient outlier nominators. *Computational Statistics and Data Analysis 34*, 279–298.

Bloedorn, E., A. D. Christiansen, W. Hill, C. Skorupka, et L. M. Talbot (2001). Data mining for network intrusion detection : How to get started. Technical report, MITRE.

Breunig, M. M., H.-P. Kriegel, R. T. Ng, et J. Sander (2000). Lof : identifying density-based local outliers. *SIGMOD Records 29*(2), 93–104.

Ceglar, A., J. F. Roddick, et D. M. W. Powers (2007). Curio : A fast outlier and outlier cluster detection algorithm for large datasets. In *2nd International Workshop on Integrating Artificial Intelligence and Data Mining*, pp. 37–45.

Chandola, V., A. Banerjee, et V. Kumar (2008). Anomaly detection - a survey. *ACM Computing Surveys To appear*, To appear.

Chimphlee, W., A. H. Abdullah, M. N. Md Sap, et S. Chimphlee (2005). Unsupervised anomaly detection with unlabeled data using clustering. In *International conference on information and communication technology*.

Daubechies, I. (1992). *Ten lectures on wavelets*. Philadelphia, PA, USA : Society for Industrial and Applied Mathematics.

Dokas, P., L. Ertoz, V. Kumar, A. Lazarevic, J. Srivastava, et P. Tan (2002). Data mining for network intrusion detection. In *NSF Workshop on Next Generation Data Mining*.

Duan, L., D. Xiong, J. Lee, et F. Guo (2006). A local density based spatial clustering algorithm with noise. In *IEEE International Conference on Systems, Man and Cybernetics*.

Eskin, E., A. Arnold, M. Prerau, L. Portnoy, et S. Stolfo (2002). A geometric framework for unsupervised anomaly detection : Detecting intrusions in unlabeled data. *Applications of Data Mining in Computer Security*, 333–342.

Ester, M., H.-P. Kriegel, J. Sander, et X. Xu (1996). A density–based algorithm for discovering clusters in large spatial databases with noise. In *2nd International Conference on Knowledge Discovery and Data Mining*, pp. 226–231.

Fan, H., O. R. Zaiane, A. Foss, et J. Wu (2006). A nonparametric outlier detection for effectively discovering top-n outliers from engineering data. In *PAKDD*.

Fujimaki, R., T. Yairi, et K. Machida (2005). An approach to spacecraft anomaly detection problem using kernel feature space. In *11th ACM SIGKDD*.

He, Z., X. Xu, et S. Deng (2003). Discovering cluster-based local outliers. *Pattern Recognition Letters 24*, 1641–1650.

Hodge, V. et J. Austin (2004). A survey of outlier detection methodologies. *Artificial Intelligence Review 22*, 85–126.

Jain, A. K. et R. C. Dubes (1988). *Algorithms for Clustering Data*. Prentice-Hall, Inc.

Jin, W., A. K. H. Tung, et J. Han (2001). Mining top-n local outliers in large databases. In *7th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 293–298.

Joshua Oldmeadow, J., S. Ravinutala, et C. Leckie (2004). Adaptive clustering for network intrusion detection. In *8th PAKDD*.

Knorr, E. M. et R. T. Ng (1998). Algorithms for mining distance-based outliers in large datasets. In *24rd International Conference on Very Large Data Bases*, pp. 392–403.

Kwitt, R. et U. Hofmann (2007). Unsupervised anomaly detection in network traffic by means of robust pca. In *International Multi-Conference on Computing in the Global Information Technology*.

Lazarevic, A., L. Ertoz, V. Kumar, A. Ozgur, et J. Srivastava (2003). A comparative study of anomaly detection schemes in network intrusion detection. In *3rd SIAM DM*.

Lee, W. et D. Xiang (2001). Information-theoretic measures for anomaly detection. In *IEEE Symposium on Security and Privacy*.

Luo, J. (1999). Integrating fuzzy logic with data mining methods for intrusion detection.

Manganaris, S., M. Christensen, D. Zerkle, et K. Hermiz (2000). A data mining analysis of rtid alarms. *Computer Networks 34*, 571–577.

Marchette, D. (1999). A statistical method for profiling network traffic. In *1st USENIX Workshop on Intrusion Detection and Network Monitoring*, pp. 119–128.

Markou, M. et S. Singh (2003). Novelty detection : a review - part 1 : statistical approaches. *Signal Processing 83*, 2481–2497.

Ng, R. T. et J. Han (1994). Efficient and effective clustering methods for spatial data mining. In *20th International Conference on Very Large Data Bases*, pp. 144–155.

Otey, M., S. Parthasarathy, A. Ghoting, G. Li, S. Narravula, et D. Panda (2003). Towards nic–based intrusion detection. In *9th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 723–728.

Papadimitriou, S., H. Kitagawa, P. Gibbons, et C. Faloutsos (2003). LOCI : fast outlier detection using the local correlation integral. In *19th ICDE*.

Patcha, A. et J.-M. Park (2007). An overview of anomaly detection techniques : Existing solutions and latest technological trends. *Comput. Networks 51*, 3448–3470.

Pires, A. et C. Santos-Pereira (2005). Using clustering and robust estimators to detect outliers in multivariate data. In *International Conference on Robust Statistics*.

Portnoy, L., E. Eskin, et S. Stolfo (2001). Intrusion detection with unlabeled data using clustering. In *ACM CSS Workshop on Data Mining Applied to Security*.

Ramaswamy, S., R. Rastogi, et K. Shim (2000). Efficient algorithms for mining outliers from large data sets. *SIGMOD Records 29*(2), 427–438.

Rousseeuw, P. et A. M. Leroy (1996). *Robust Regression and Outlier Detection*. Wiley-IEEE.

Schölkopf, B., J. Platt, J. Shawe-Taylor, A. Smola, et W. R. (2001). Estimating the support of high-dimensional distribution. *Neural Computation 13*, 1443–1471.

Tax, D. M. J. et R. P. W. Duin (2001). Combining one–class classifiers. *Lecture Notes in Computer Science 2096*, 299–317.

Valdes, A. et K. Skinner (2001). Probabilistic alert correlation. In *Recent Advances in Intrusion Detection*, pp. 54–68.

Vinueza, A. et G. Grudic (2004). Unsupervised outlier detection and semi–supervised learning. Technical Report CU-CS-976-04, Univ. of Colorado at Boulder.

Wu, N. et J. Zhang (2003). Factor analysis based anomaly detection. In *IEEE Workshop on Information Assurance*.

Yegneswaran, V., P. Barford, et S. Jha (2004). Global intrusion detection in the domino overlay system. In *Network and Distributed Security Symposium*.

Zhong, S., T. M. Khoshgoftaar, et N. Seliya (2007). Clustering-based network intrusion detection. *International Journal of Reliability, Quality and Safety Engineering 14*, 169–187.

Summary

La détection d'intrusion est un domaine important pour la sécurité des systèmes d'information. Les systèmes de détection d'intrusion (IDS) les plus utilisés reposent sur la détection de signatures et ont besoin de mises à jour fréquentes pour défendre un système contre les nouvelles attaques. D'un autre côté, la détection d'anomalie peut compenser ce besoin, mais provoque de nombreuses fausses alarmes. En effet, un comportement qui dévie de manière significative des comportements habituels sera considéré comme dangereux par un IDS utilisant les anomalies. Isoler les véritables intrusions dans un ensemble d'alarmes est donc un défi important pour tout IDS. Dans cet article, nous considérons une nouvelle caractéristique pour isoler les comportements malicieux. Cette caractéristique est basée sur leur possible répétition d'un système d'information à un autre. Nous proposons un nouveau principe de détection des objets atypiques et le validons par une série d'expérimentations.