Monitoring the Network Monitoring System: Anomaly Detection using Pattern Recognition

Maha Mdini, Alberto Blanc, Gwendal Simon
IMT Atlantique, France
{firstname.lastname}@imt-atlantique.fr

Jérôme Barotin, Julien Lecoeuvre
Astellia, France
{j.barotin, j.lecoeuvre}@astellia.com

***Abstract*—For a successful and efficient network supervision, an Anomaly Detection System is essential. In this paper, our goal is to develop a simple, practical, and application-domain specific approach to identify anomalies in the input/output data of network probes. Since data are periodic and continuously evolving, it is not possible to use threshold-based approaches. We propose an algorithm based on pattern recognition to help mobile operators detect anomalies in real time. The algorithm is unsupervised and easily configurable with a small number of tuning parameters. After weeks of deployment in a production network monitoring system, we obtain satisfactory results: we detect major anomalies with low error rate.**

I. Introduction

Mobile operators exploit Anomaly Detection Systems (ADSs) in their networks to detect and report any anomaly as fast as possible. Since mobile networks are getting more complex, ADSs have become more prone to failures, which can lead to a Quality of Experience (QoE) degradation for the customers. Monitoring the network monitoring system is thus critical to ease network troubleshooting.

Our goal is to assist network administrators in the task of investigating failures. Since both the network and the monitoring system are complex, determining whether the root cause of a reported anomaly is localized in the monitoring system or in the cellular network is not straightforward. The investigation task is hard and time consuming. For this reason, we worked on automating the ADSs monitoring.

We discuss in Section II the existing anomaly detection algorithms, but none of them addresses all the requirements that are specific to a monitoring systems. First, the ADS must use few computational resources. Second, it must not be perturbed by periodic variations or by the normal increase in traffic volume. Third, it must have few configuration parameters. This last requirement is essential for systems that are meant to be used in production environments.

In this paper, we propose an ad-hoc solution to the network anomaly detection problem, called Watchmen Anomaly Detection (WAD). WAD runs in real time on data coming from monitoring devices, such as network probes measuring traffic throughput on network interfaces. WAD applies a specific transformation highlighting abrupt changes to these measurements. Then, based on this transform, it creates reference patterns describing data in normal state. Afterwards, it measures the gap between the reference pattern and real time data. If the gap is large, the administrator is notified about an anomaly occurring in the related equipment. For example, a network administrator can easily diagnose a Mobility Management Entity (MME) overload due to a specific event involving a large number of people (e.g., concert, sport match) by being alerted of an abnormal rise of the number of Call Data Records (CDRs) in the input of the probe monitoring this MME.

The paper is organized as follows: We present the state of the art in Section II. In Section III, we explain the mathematical basis and the architecture of WAD. Then, we demonstrate the robustness of our solution and compare it with some reference techniques. We conclude with our main results and a brief discussion of future work.

II. Related Work

Classification versus Regression Problems When it comes to anomaly detection, one may think that this issue is a classification problem. Anomalies can been seen as points belonging to an outlying cluster. In this case, one can use k-means and related algorithms [1, 2]. Since partitioning data into an anomaly-free cluster and an anomalous one is not always possible [3], k-means is often paired with decision trees [2]. And as the decision tree rules are based on the results of the k-means clustering, this can lead to an unacceptable global mis-classification.

In the case of time series, the Symbolic Aggregate Approximation (SAX) algorithm [4] can be an efficient solution, however, it has some limitations when applied to periodic data where certain patterns considered as normal in a given part of the period can be considered as anomalies in another one. SAX, which "does not keep track of time", does not detect these anomalies. That is why, in our solution, we maintain our focus on the temporal dimension of data.

Anomaly detection in time series can also be addressed as a regression problem, by first determining an acceptance interval for future values and then by declaring as anomalies all the points outside the interval. Autoregressive Integrated Moving Average (ARIMA) models are accurate regression algorithms, which have been successfully used for anomaly detection [5, 6]. However, these models require significant computational resources, which is not acceptable in highly dynamic systems such as network ADSs. Similarity, we created a predictive model but unlike ARIMA, WAD implements a simple data transformation that is computationally inexpensive.

Another approach to the anomaly detection is based on Principal Component Analysis (PCA). Lakhina et al. [7] showed how PCA helps detecting peaks and dips in noisy data. However PCA remains a linear decomposition and requires the noise to be linearly separable from significant data, condition which is not always satisfied [8].

Temporal versus Spectral Approach Anomalies can be defined as abrupt changes in time series. Such changes are reflected as high frequencies components. Thus some studies suggest to use spectral analysis to detect anomalies [9, 10]. Han et al. [9] showed how to detect anomalies using clustering techniques on the Fast Fourier Transform (FFT) of the time series. The main weakness of FFT is when one has to find the time at which each frequency occurs. Thus, the wavelet transform can be a better choice. Barford et al. [10] explained how a static threshold is enough for anomaly detection once one applies the wavelet transform. Unlike [9, 10], we implemented a local transform that involves only few samples. Some other works go beyond FFT and Wavelet transform, applying more signal processing transformations such as Hough transform [11] and Hilbert transform [12]. A number of papers deal with raw data and highlight temporal aspects of data rather than spectral ones [4, 13]. This approach does not work for periodic data since a normal value at some point of the period can be an anomaly at another point.

III. Watchmen Anomaly Detection

The WAD algorithm detects anomalies in a cellular network monitoring system, by analyzing the traffic generated by the monitoring system itself. We illustrate the overall system in Figure 1. Each monitored entity generates CDRs, which are sent to the *system monitoring* module of the ADS, and Record Generation Reports (RGRs) to WAD. RGRs are WAD reports generated from monitor system metrics. RGRs include the number of CDRs, TCP Data Records (TCPDRs), the sessions/bearer deciphering rate, etc. The stream of data logs is a univaraite time series. Since the data come from the network probes, they are highly correlated with the subscribers behavior (e.g., they show a daily pattern). In the remainder of the paper we use a period of one day but WAD works with any period length. The goal of WAD is to detect anomalies

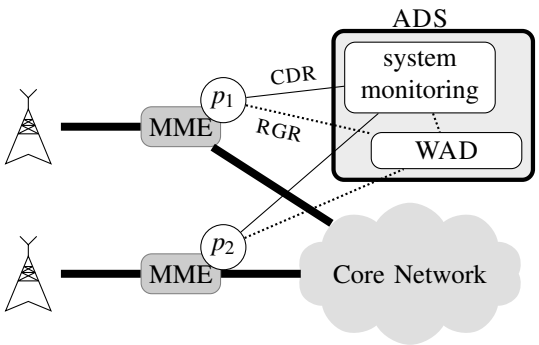

Fig. 1: Overall Architecture: two probes p_1 and p_2 monitor MMEs and send reports to the ADS

in time series such as peaks, dips and sudden rises and falls. Those anomalies are caused by erroneously configured devices, broken equipment, and atypical events (e.g., traffic increase due to a sporting event), just to name a few. Figure 2 shows the types of anomalies that we want to detect.

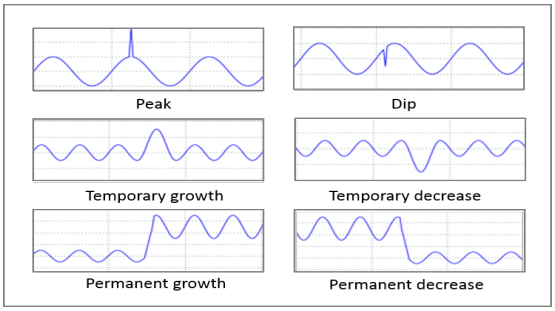

Fig. 2: The different anomaly types

WAD consists of two phases: a learning phase and a detection phase. The learning part is executed once a day (i.e., the period of the data). In this phase, we create a reference model, which is stored in a database. This reference model is then used in the detection phase, which runs in real time.

A. Learning Phase

Figure 3 shows the four main steps of the learning phase, whose input is one month of data for each metric. Note that WAD handles each metric independently of the others. Future work will cover WAD optimization based on metrics correlation.

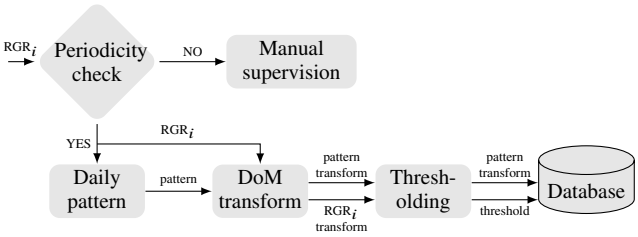

Fig. 3: The learning phase for a metric RGR$_i$

1) Periodicity Check: We compute the FFT of the time series to check whether the data is periodic, in which case there is dominant frequency in the spectrum. To verify this, we calculate the ratio between the sum of the components close to the dominant frequency and the sum of the rest of the components. The ratio should be greater than one otherwise the noise mask the periodicity. Based on empirical tests, we took five as the lower bound of the ratio. If a given metric is not periodic, we ignore it for a day. This metric will be checked manually by network administrators. This can happen on freshly installed systems.

2) Daily Pattern: In order to obtain the anomaly free behavior, we compute a daily pattern for each metric. To do so, we split our training set into periods and calculate the average value on each point of the period. In practice, we have one

sample every 15 minutes. We compute the average for all the days of the month at that time. This way, we get a vector of values describing the metric average evolution.

We then calculate the Euclidean distances between the average vector and the periods of the training set. We sort these distances and discard all the periods whose distances to the average vector are greater than the 95th percentile. This way, we rebuilt a training set with no extreme values. Afterwards, we calculate the average vector of all the periods of the anomaly-free set. This vector is the metric daily pattern.

*3) Difference over Minimum (*DoM*) Transform:* The data we use for the model generation have to be smooth, since discontinuities are considered to be anomalies. For this reason, we need a transform that amplifies jumps and minimizes smooth variations. To this end, we define the DoM transform (T) of a function f as:

$$T(f, L) = d\left(f\left(t\right), f\left(t - L\right)\right)$$

where d is the continuous extension of f defined as:

$$f(x, y) = \frac{x - y}{\min(x, y)}$$

and the lag $L > 0$ is the order of the transform. This transform quantifies the jumps in f after L time steps, starting from a given time t. In the remainder of this paper, we focus on the transform of first order since it acts like a derivative and it features that it goes to infinity when there is a peak (or a dip) and it sticks to small values otherwise.

4) Thresholding: At this point of the algorithm, we measure the distance between the transformed value of each sample of the metric and the corresponding (with regard to the time of the day) transformed value of the pattern. We sort these distances and we set the threshold to the 99th percentile. We store the threshold and the DoM transform of the daily pattern in the database for real time detection.

B. Detection Phase

Figure 4 shows the three main steps of the detection phase, which runs in real time and compares the incoming sample to the reference model constructed in the previous phase.

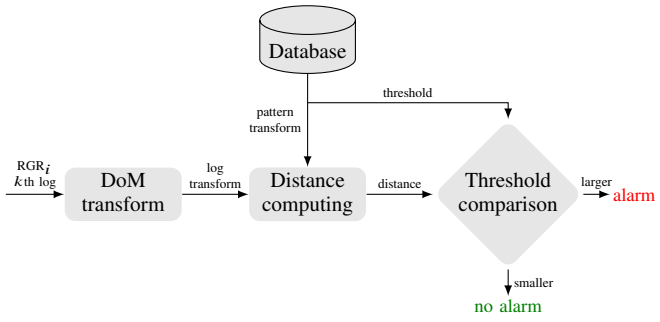

Fig. 4: Detection phase for the kth sample of the RGR$_i$

After computing the DoM transform of a new sample, we calculate the Euclidean distance to the reference pattern for the same time of the day. If the distance is larger than the threshold, plus a tolerance band, WAD raises an alarm. The tolerance band reduces the number of false positives, as there can be occasional fluctuations that cross the threshold without being anomalies. Based on empirical observations, we set the tolerance band to 0.1.

IV. Validation Tests

The goal of WAD is to be a lightweight solution to detect anomalies in the monitoring system of a cellular network. We report in the following some of our main findings after a series of evaluations, first in the lab based on real data traces, and then after we installed WAD in the production network of two operators.

A. Accuracy

1) Lab results: We tested WAD in the lab based on historical data from an European operator. We worked with the values of 58 metrics over a month containing 90 anomalies. We compared our algorithm with two reference solutions: SAX and PCA. We chose SAX and PCA since they fit our use case: they are easy to implement and have modest computational requirements as our solution. We have used the recommended parameters for each algorithm. For SAX, we used an alphabet size of 15 and a word size of 4 [4]. We used the PCA decomposition with four-dimensional original and projection spaces. We detected anomalies on the fourth component and we used a threshold equal to the 99.5th percentile of the data [7].

We evaluated the results in terms of True Positives (TP), False Positives (FP) and False Negatives (FN). The TP are anomalies detected by the system. The FP are normal points erroneously labeled as anomalies. The FN are anomalies not detected by the system. We also measured the sensitivity, which is the ratio of correctly detected anomalies to the total number of anomalies, and the specificity, which is the ratio of points correctly identified as normal to the total number of normal points. Table I shows the results of these tests.

Algorithm	TP	FP	FN	Sensitivity	Specificity
SAX	59	6	31	0.65	0.99
PCA	85	85	5	0.94	0.99
WAD	81	9	9	0.90	0.99

TABLE I: Lab Evaluation Results

The results show that the PCA is a good method to detect abrupt changes. However, including anomalies in the noise component implies a high FPs rate. The SAX algorithm obtains a high FN number since SAX does not keep track of time. Thus, it considers some behaviors as being normal although they abnormally took place at an unusual time.

2) Production Network Results: We run WAD for three months to supervise the network of two mobile operators (in Europe and Africa). The first operator gave a general positive feedback, without giving specific numbers. The second one

gave us access to a detailed feedback. In the remainder of this section, we will refer to these results.

The network administrator selected 18 metrics and provided feedback about the performance of our algorithm. During the three months, 83 anomalies occured (among 155 520 samples). We detected 75 of them, which means we had 8 FN. We also detected four FP. The sensitivity of the system is equal to 0.90 and the specificity is equal to 0.99. The accuracy of WAD meets the requirements of both operators.

B. Memory and Computation Needs

WAD has a $O(nlog(n))$ complexity, as the FFT is the term with the dominant time complexity. To evaluate its hardware performance, we installed WAD on a machine with the following features: 4 Cores, 4GB RAM, 30GB hard drive. The learning phase takes about 30 minutes to process 850MB of data. Tables II shows the results of these tests.

	CPU		**Memory**	
	Median	**Max**	**Median**	**Max**
Learning Phase	12%	25%	1.2%	1.6%
Detection Phase	0.1%	2%	0.5%	0.9%

TABLE II: CPU and Memory Usage

Based on these experiments and on the successful deployment in production networks, we conclude that WAD has modest requirements in terms of memory and CPU. Concerning the scalability aspect, WAD has low computational complexity and treats metrics independently. These facts make WAD easy to deploy on Big Data infrastructures.

C. Discussion

WAD gives promising results in terms of gain of productivity, time saving, and ease of outage troubleshooting. After using WAD for a few months, the network administrators of two cellular operators have highlighted the following aspects. They have noticed that they do not have to check all the metrics manually, since they are alerted in real time. They also save a lot of time and effort in troubleshooting and understanding issues: WAD helps them localize the problem and therefore fix broken devices and/or configurations efficiently. The configuration of our solution is straightforward since it produces dynamic models and does not require calendar integration. Finally, as an added benefit, it does not require a large training set. Nevertheless, our solution still has some configurability issues that can be improved: The accuracy of WAD relies on the tolerance band value and the pattern calculation percentiles. Although we obtained satisfactory results on different data sets with the same parameters, we need to find a way to automatically compute parameters based on the variance of the data.

V. Conclusion and Future Work

Automatic supervision of network monitoring systems and real time feedback processing is crucial to automate mobile network troubleshooting. Thus, integrating an ADS based on Machine Learning techniques, like WAD, within a network monitoring system seems to be an essential task for the creation of self healing networks. Our research may contribute to the definition of basic functions for self-organizing and more resilient networks.

We are working on improving WAD by using the Hurst exponent to automatically compute the error term (tolerance band). We also plan to improve the data transformation. The DoM transform is finely tuned to amplify the type of anomalies we want to detect for network monitoring. Ideally, we would like to be able to automatically generate data transformation operations that can amplify given events.

Bibliography

[1] G. Mnz, S. Li, and G. Carle, "Traffic Anomaly Detection Using KMeans Clustering," in *Proc. of In GI/ITG Workshop MMBnet*, 2007.

[2] A. P. Muniyandi, R. Rajeswari, and R. Rajaram, "Network Anomaly Detection by Cascading K-Means Clustering and C4.5 Decision Tree algorithm," *Procedia Engineering*, vol. 30, 2012.

[3] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection: A Survey," *ACM Comput. Surv.*, vol. 41, no. 3, 2009.

[4] E. J. Keogh, J. Lin, and A. W. Fu, "HOT SAX: efficiently finding the most unusual time series subsequence," in *Proc. of IEEE ICDM*, 2005.

[5] H. Z. Moayedi and M. A. Masnadi-Shirazi, "Arima model for network traffic prediction and anomaly detection," in *Proc. of IEEE ISCIT*, vol. 4, 2008.

[6] E. H. M. Pena, M. V. O. de Assis, M. Lemes, and P. Jr, "Anomaly detection using forecasting methods arima and hwds," in *Proc. of IEEE SCCC*, 2013.

[7] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," in *Proc. of ACM SIGCOMM*, 2004.

[8] H. Zou, T. Hastie, and R. Tibshirani, "Sparse Principal Component Analysis," *Journal of Computational and Graphical Statistics*, vol. 15, no. 2, 2006.

[9] T. Han, Y. Lan, L. Xiao, B. Huang, and K. Zhang, "Event detection with vector similarity based on fourier transformation," in *Proc. of IEEE ICCSSE*, 2014.

[10] P. Barford, J. Kline, D. Plonka, and A. Ron, "A signal analysis of network traffic anomalies," in *Proc. of ACM SIGCOMM Workshop IMW*, 2002.

[11] R. Fontugne and K. Fukuda, "A hough-transform-based anomaly detector with an adaptive time interval," in *Proc. of ACM SAC*, 2011.

[12] A. Subbu and A. Ray, "Space partitioning via Hilbert transform for symbolic time series analysis," *Applied Physics Letters*, vol. 92, no. 8, 2008.

[13] M. Thottan and C. Ji, "Anomaly detection in IP networks," *IEEE Trans. Signal Processing*, vol. 51, no. 8, 2003.