2013 IEEE 25th International Conference on Tools with Artificial Intelligence

An Intelligent Anomaly Detection and Reasoning Scheme for VM Live Migration via Cloud Data Mining

Qiannan Zhang, Yafei Wu, Tian Huang, Yongxin Zhu
School of Microelectronics
Shanghai Jiao Tong University
Shanghai, China
{zhangqiannan,wuyafei,huangtian}@ic.sjtu.edu.cn, zhuyongxin@sjtu.edu.cn

***Abstract*—Cloud computing operators provide flexible, convenient, and affordable means to access public and private services. Virtual machine (VM) live migration, as an important feature of virtualization technique in cloud computing, ensures high efficiency and performance of computing infrastructure, while it stays transparent to clients. However, VM live migration is observed to cover anomalies due to their statistical similarity. To tackle the critical security issue, in this work, we propose an intelligent scheme to mine statistical data from cloud infrastructure to detect anomalies even if VMs are migrated to a new host with different infrastructure settings. In addition to detection of the existence of anomalies, our scheme is capable of identifying the possible sources of anomalies, which gives administrators clues to pinpoint and clear the anomalies.**

Keywords- VM live migration, anomaly detection, anomaly reasoning, cloud computing, SAX, LOF

I. INTRODUCTION

As the reappearance of utility computing paradigm, cloud computing systems deliver massively scalable IT-enabled capabilities as a service to clients using internet technologies [1]. Although cloud computing systems, together with virtualization techniques, have become mature and commercialized recently, they are still vulnerable to hackers' attacks as well as operational errors. Statistics show that Gmail servers broke down four times in December, 2012. Since 2007, 13 famous cloud computing servers collapsed for 568 hours, which leads to $71,700,000's economic loss. Those safety issues are caused by many reasons: vicious intrusion, misusage or hardware breakdown. The diversity of anomalies makes traditional detection methods difficult to be comprehensive, let alone categorizing anomalies.

In the field of cloud computing, anomalies are generally of two kinds: collective anomalies and contextual anomalies [2]. Collective anomaly happens when the combination of several elements are abnormal, although any one of them acts regularly. E.g., either the fact that the number of I/O request is high or I/O transmission rate is low can be normal. However, if I/O request is frequent and its transmission rate is low, anomaly may be reflected. On the other side, contextual anomaly happens with certain context. It is difficult to tell whether a data point is a contextual anomaly or not without analyzing the other data points around it. E.g., for an SNS (Social Networking Services) server, high CPU usage is normal in daytime while abnormal during the night.

As an underlying technique of virtualization featuring efficient cloud computing systems, virtual machine (VM) live migration refers to the process of moving a running virtual machine or application between different physical machines without disconnecting the client or application. As a result of VM migration's merits of balancing the workload of physical machines, elastic scaling, fault tolerance, hardware maintenance, sharing infrastructures and keeping mechanism transparent to users, this technique is widely used in reality. Unfortunately, VM live migration adds difficulties to anomaly detection since it is actually based on large numbers of memory copy operations, which often cover anomalies and attacks. In additional, anomaly detecting becomes even harder when VMs are migrated to a destination host whose workload, memory size, network condition and other infrastructure settings are quite different from the origin host, so, applications may behave differently.

In this paper, we challenge attacks from hackers and errors by careless operators by enabling anomaly detection during VM live migration. Data mining methodology is implemented by giving machines intelligence to learn the regular pattern and tell us what is wrong and where the problem is. E.g., if a VM is migrated from a busy server to one with less workload, it is reasonable that CPU idle time will increase and it cannot be regarded as an exception.

In this work, we propose an intelligent scheme for anomaly detection and reasoning on cloud computing based on resources utilization statistics. Our major contributions are as follows. Firstly, adaptive LOF (Local Outlier Factors) is proposed to make the classic LOF algorithm still valid when smooth environmental changes happen. Secondly, we further extended LOF [3] algorithm to reveal the type of anomaly. Dimension reasoning LOF (DR-LOF.) can point out the "most anomalous" dimension of data. Thus, the administrators can properly take action to hit back the intrusion or fix the vulnerability. As far as we know, no existing solution focuses on anomaly reasoning in cloud computing. Thirdly, a big advantage of our intelligent anomaly detection scheme lies on the fact that the protected applications break the limit of running on the same host with stable infrastructural conditions. With the implementation of adaptive DR-LOF and SAX (Symbolic Aggregate ApproXimation) [4], the intelligent scheme is capable of understanding time information and detecting contextual anomalies. After VM is migrated to the destination host or experienced other environmental changes, the detection scheme is still accurate and effective.

1082-3409/13 $31.00 © 2013 IEEE
DOI 10.1109/ICTAI.2013.68

The rest of the paper is organized as follows. Section Ⅱ discusses the related work and the background of classic LOF and SAX algorithms. Section Ⅲ gives a scheme overview and in Section Ⅳ novel extensions to former algorithms is proposed and implemented into anomaly detection scenario in cloud computing. Section Ⅴ shows the evaluation results and compare our scheme with classic LOF and clustering. Finally, conclusion is drawn in Section Ⅵ.

II. Related Work and Background

A. Related work

As a classic issue, anomaly detection has been explored by researchers in the areas of intrusion detection, machine learning as well as relevant statisticians. The problem arose again with new technical and operational features in cloud computing systems, though there have been well formed methods for operations on classic computer systems.

Many literatures on intrusion detection draw on the experience of machine learning. Classification, clustering and Markov chain [5] [6] and are well studied for data centers, while anomaly detection methodology in cloud computing system is still in its early stage. Since training phase will settle down the dividing lines between normal and abnormal data, current machine learning methods used for anomaly detection in cloud computing is highly dependent on stable infrastructure settings. Although some unsupervised algorithms do not have training phase, they still have something to do with previous experience, which is not appropriate for VM migration scenarios.

Among methods handling anomaly detection in cloud systems, there are a few efforts on security issues in VM live migration. [7] proposed a role-based mechanism with remote attestation to leverage Intel vPro and TPM to improve security. VM migration is controlled by specific policies that are protected in seal storage under the mechanism. [8] built a prototype system of the framework based on stateful firewall enabling Xen hypervisor. [9] focused on feature selection algorithm to enhance detection accuracy in VM, but it did not consider VM migration scenario. Most of these works focused on generic frameworks to keep VM from attacks but none of them gave feasible ways to detect and fix anomalies after attacks.

In previous work [10], we proposed to implement LOF for anomaly detection on cloud computing. The algorithm can successfully solve the task when the environment is stable. However, if VM migration is performed, it no longer meets the requirements. In our experiments, LOF generates huge amount of false alarms and misses some anomalies on the destination host after the introduction of VM migration. The concept of viewing data as separated points in feature space loses time information. As a result, LOF is weak in understanding the relationships between two successive points and mistakes some normal behaviors for anomalies.

B. Background

Our algorithms are developed based on LOF and SAX. We will give a brief description of the basic algorithms proposed by other researchers here and our extended algorithms in Section IV. Our extended algorithms fit cloud computing better and can dig out more useful information.

1) LOF

LOF was raised by Markus M. Breunig et al. [3]. It over-performs unsupervised SVM, Mahalanobis approach and NN approach and many other anomaly detection algorithms [11]. So, we choose it as our basic strategy. To search nearest neighbors in LOF, we implement NNS (Nearest Neighbor Search) [12] algorithm. R-tree [13] is chosen to keep raw data structured as it is good at handling spatial data [4].

Suppose $kdistance(A)$ is the distance between point A and A's the k^{th} nearest neighbor. $N_k(A)$ is the set of A's k nearest neighbors and $|N_k(A)|$ is the size of the set, which equals to k. $dist(A,B)$ represents the Euclidean distance between A and B. Reachability distance between A and B, $rd_k(A,B)$, is defined as:

$$rd_k(A,B) = \max\{kdistance(A), dist(A,B)\} \quad (1)$$

Local reachability density, $lrd_k(A)$, is defined as:

$$lrd_k(A) = \frac{1}{\frac{\sum_{B \in N_k(A)} rd_k(A,B)}{|N_k(A)|}} \quad (2)$$

LOF value, $LOF_k(A)$, is defined as:

$$LOF_k(A) = \frac{\sum_{B \in N_k(A)} \frac{lrd_k(B)}{lrd_k(A)}}{|N_k(A)|} \quad (3)$$

The concept of relative density makes the algorithm fairer [3]. If LOF value is higher than certain threshold, point A will be regarded as an anomaly.

2) SAX

SAX is a symbolic representation of time series. It is introduced by E. Keogh et al. [4]. Tons of methods have been designed to deal with time sequences in the past decades, such as DFT (Discrete Fourier Transformation) and DWT (Discrete Wavelet Transformation), but they are not so good as SAX because of their inferior performance and complexity [15].

The process of SAX consists of three phases: Normalization, Symbolization and Similarity Comparison. Time series is firstly normalized to Gaussian distribution with zero mean and variance equaling to 1, *N(0,1)*. In Symbolization phase, PAA algorithm (Piecewise Aggregate Approximation) is used for dimension reduction: a time series is divided into several pieces and each piece is represented by its mean. Then, to obtain discrete representation, a statistic look-up-table is used to map values into alphabetic letters. Since the sequence follows Gaussian distribution, the thresholds are set as the cutoffs that make the area under Gaussian curve *N(0,1)* equalized. Finally, symbolic distance is calculated as follows. Suppose:

P, Q: two time series;

p_i, q_i: the i^{th} element in time series;

$\tilde{P}$, $\tilde{Q}$: corresponding symbolic series;

$\tilde{p}_i$, $\tilde{q}_i$: the i^{th} element in symbolic series.

The symbolic distance between $\tilde{P}$ and $\tilde{Q}$ is defined as:

$$SymDist(\tilde{P},\tilde{Q}) = \sqrt{\frac{n}{w}}\sqrt{\sum_{i=1}^{w} dist^2(\tilde{p}_i,\tilde{q}_i)} \quad (4)$$

Where dist() is defined as:

$$dist(a,b) = b_{\max(a,b)-1} - b_{\min(a,b)} \text{ when } |a-b|>1 \quad (5)$$

dist() is not Euclidean distance since the symbolic distance of the letters should be the lower bound of Euclidean distance of raw time series [15]. The detailed definitions, deductions and look-up-table can be found in [4].

SAX is utilized as a tool of subsequence matching: comparing the index of similarity before and after VM migration.

III. SCHEME OVERVIEW

The intelligent anomaly detection and reasoning scheme manipulates the statistical data sampled from cloud platform to discover the hostile attacks and report the anomalies with their possible sources to the administrators. Our AI scheme's framework is shown in Fig. 1.

We firstly apply adaptive DR-LOF algorithm to detect collective anomalies and figure out the dimension of statistical data with the most significant impacts on the abnormal condition. Caused by misinterpret of contextual information, this temporary result may contain unexpected false alarms. Further re-validation processes will be performed if the percentage of alarms exceeds the threshold generalized by former experience, which means that it is likely that the application is staying in a new environment (e.g. VM migration). Thus the scheme deals with contextual anomalies using SAX algorithm: comparing the symbolic distances of data before and after live migration to tell whether applications operate regularly in new environment.

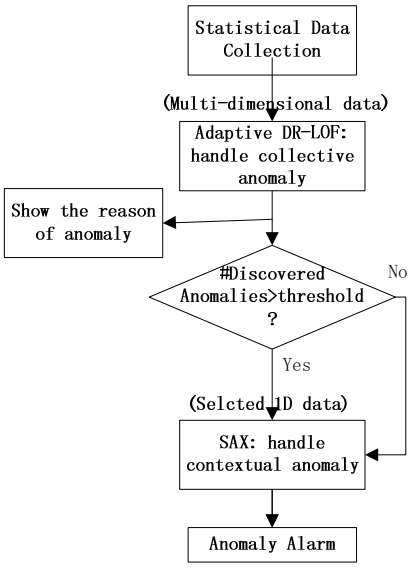

Figure 1. Intelligent anomaly detection and reasoning scheme overview.

IV. ALGORITHMS AND IMPLEMENTAION

A. Data Collection

SAR (System Activity Reporter) is used to monitor and collect statistical data via Linux platform. It can monitor a lot of system activities, such as file read/write, system call, CPU usage, disc I/O and so on. The statuses of those resources' utilization are displayed in a time-stamped report.

In our experiment, 38 statistical indicators from SAR are chosen to form the basic database.

B. Adaptive LOF

Served as knowledge base, R-tree is fixed during training phase, hence the changes caused by migration or other factors will make the results meaningless. Therefore, the method proposed by E. Keogh et al. in [4] is insufficient to handle the VM migration scenario discussed in this paper.

To explain our scheme clearly, let us consider a 2-dimensional case for simplicity. In Fig. 2, suppose the normal behaved cluster moves slowly from *C1* to *C2* due to certain environmental reasons. In this case, if we do not update knowledge base and deem *C1* as the normal cluster from beginning to end, *p1* will be considered as a regular data point while *p2* will not be. However, on the contrary, since the normal behaviors have moved to *C2*, *p2* is actually the normal point while *p1* is not. Moreover, updating is even more important in our migration scenario since the infrastructure settings may change a lot during migration.

Our experiments show that without updating, if data volume grows linearly or exponentially (e.g. *D1 = {1, 2, 3,…}* or *D2 = {1, 10, 100, 1000,…}*), classic LOF value increases rapidly to infinity. Nevertheless, if we update R-tree every time we figure out a new normal testing point, LOF will converge to a certain finite value.

To avoid potential instability, more operations are needed: Data should be normalized before it is put into R-tree. As the normalization rule is generated in training phase, it may no longer be valid for testing data. So, every time a new data record is put in, inspection should be done to all the multi-dimensional data in R-tree to check if their outlines are still similar to a hypercube. If not, the entire R-tree should be re-normalized and rebuilt.

The overall LOF section of anomaly detection is summarized in Fig. 3.

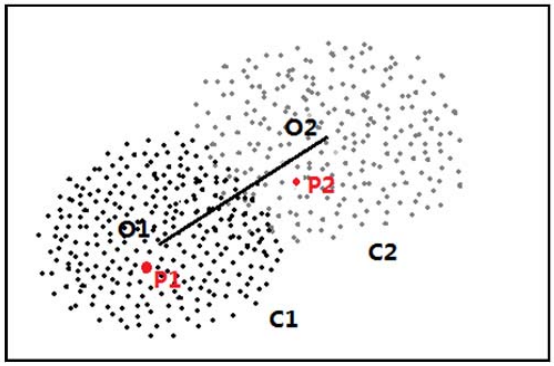

Figure 2. Illustration on normal behaved cluster moving process generated by environmental changes.

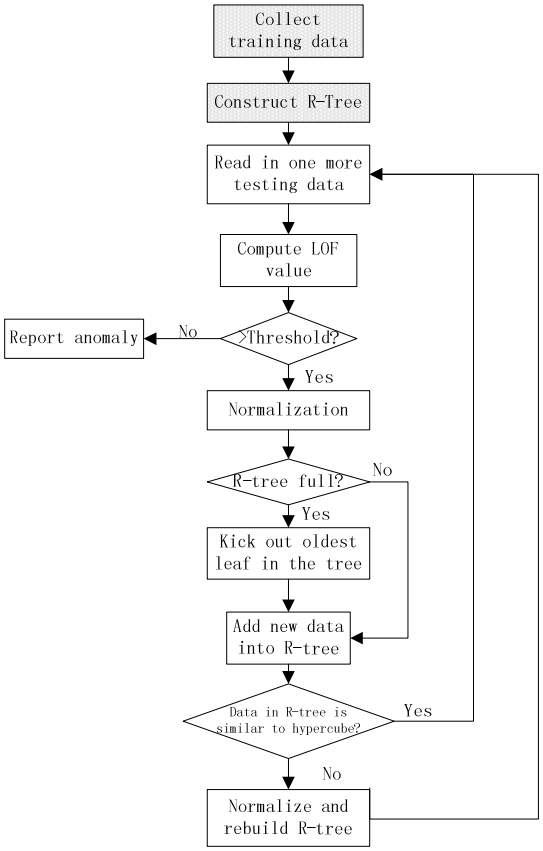

Figure 3. Flow chart of adaptive LOF (Grey rectangles represent training phase, White rectangles represent test phase).

C. DR-LOF

The basic idea of DR-LOF is that by removing one dimension data of each data point and recalculating LOF, the ratio of the new LOF value to total LOF value reflects the status of both this data point and the removed dimension of data. Here, statement and demonstration of these properties are given below. Variable with superscript n represents that it is computed without the n^{th} dimensional data.

With Eq. (1) (2) (3), the ratio of LOF values without and with the n^{th} dimensional data can be represented as:

$$\frac{LOF_k^{\,n}(A)}{LOF_k(A)} = \frac{\dfrac{\sum_{B\in N_k(A)} \dfrac{lrd_k^{\,n}(B)}{lrd_k^{\,n}(A)}}{|N_k(A)|}}{\dfrac{\sum_{B\in N_k(A)} \dfrac{lrd_k(B)}{lrd_k(A)}}{|N_k(A)|}} \tag{6}$$

Supposing C is the set of points of B's k nearest neighbors and according to Eq. (2), we have:

$$\frac{LOF_k^{\,n}(A)}{LOF_k(A)} = \frac{\sum_{B\in N_k(A)} \dfrac{\sum_{B\in N_k(A)} rd_k^{\,n}(A,B)}{\sum_{C\in N_k(B)} rd_k^{\,n}(B,C)}}{\sum_{B\in N_k(A)} \dfrac{\sum_{B\in N_k(A)} rd_k(A,B)}{\sum_{C\in N_k(B)} rd_k(B,C)}}$$

$$= \frac{\sum_{B\in N_k(A)} rd_k^{\,n}(A,B)}{\sum_{B\in N_k(A)} rd_k(A,B)} \times \frac{\sum_{B\in N_k(A)} \dfrac{1}{\sum_{C\in N_k(B)} rd_k^{\,n}(B,C)}}{\sum_{B\in N_k(A)} \dfrac{1}{\sum_{C\in N_k(B)} rd_k(B,C)}} \tag{7}$$

Based on Eq. (7), we can obtain the following properties.

Property 1. *If data point A is normal, removing any dimension of A and re-compute LOF with the other dimensions of data will get approximately the same LOF value.*

Prop. 1 can be explained by making two assumptions first for simplicity: (1) all A's k nearest neighbors are on the hyper-sphere centered at A with radius of $kdistance(A)$; (2) the distances between A and any of its k nearest neighbors on any dimension are the same, say a. Since data has been normalized, these two assumptions are fully acceptable. So, reachability distance between A and B is defined as:

$$rd_k(A,B) = \sqrt{(A_1-B_1)^2+(A_2-B_2)^2+...+(A_N-B_N)^2} = \sqrt{a^2+a^2+...+a^2} = a\sqrt{N} \tag{8}$$

A_n and B_n are A and B's projection on the n^{th} dimension coordinate. Similarly, $rd_k^{\,n}(A,B) = a\sqrt{N-1}$. N is the dimensionality of the data. Eq. (7) can be further deduced to:

$$\frac{LOF_k^{\,n}(A)}{LOF_k(A)} = \frac{\sum_{B\in N_k(A)} a\sqrt{N-1}}{\sum_{B\in N_k(A)} a\sqrt{N}} \times \frac{\sum_{B\in N_k(A)} \dfrac{1}{\sum_{C\in N_k(B)} a\sqrt{N-1}}}{\sum_{B\in N_k(A)} \dfrac{1}{\sum_{C\in N_k(B)} a\sqrt{N}}}$$

$$= \frac{ka\sqrt{N-1}}{ka\sqrt{N}} \times \frac{\dfrac{k}{ka\sqrt{N-1}}}{\dfrac{k}{ka\sqrt{N}}} = \sqrt{\frac{N-1}{N}} \times \sqrt{\frac{N}{N-1}} = 1 \tag{9}$$

Property 2. *If data point A is an anomaly:*

i. *LOF value without the n^{th} dimensional data decreases if the n^{th} dimension is anomalous;*
ii. *LOF value without the n^{th} dimensional data increases if the n^{th} dimensional data behaves normally.*

The explanation of Prop. 2 follows here. Since A is anomalous, it is away from the normal behaved cluster C. Suppose A's projections on u of N dimensions coordinates fall into cluster C and v dimensions do not ($u+v=N$), and all the distances from any of A's normal dimensions to any point in cluster C are a while those of A's abnormal dimensions are b, where $b \gg a$. So, similar to Eq. (8), $rd_k^{\,n}(A,B) = \sqrt{a^2u+b^2v}$.

1) The removed n^{th} dimension data is anomalous.

To prove corollary i, remove the n^{th} anomalous dimension data to have $rd_k^{\,n}(A,B) = \sqrt{a^2u+b^2(v-1)}$.

$$\frac{LOF_k^{\,n}(A)}{LOF_k(A)} = \sqrt{\frac{a^2u+b^2(v-1)}{a^2u+b^2v}} \times \sqrt{\frac{N}{N-1}} \tag{10}$$

As $b \gg a$,

$$\frac{LOF_k^{\,n}(A)}{LOF_k(A)} \approx \sqrt{\frac{v-1}{v}} \times \sqrt{\frac{N}{N-1}} \le 1 \tag{11}$$

2) The removed n^{th} dimension data is normal.

To prove corollary ii, remove the n^{th} anomalous dimension data to have $rd_k^n(A,B)=\sqrt{a^2(u-1)+b^2v}$. Since $b \gg a$,

$$\frac{LOF_k^n(A)}{LOF_k(A)}=\sqrt{\frac{a^2(u-1)+b^2v}{a^2u+b^2v}}\times\sqrt{\frac{N}{N-1}}\approx\sqrt{\frac{N}{N-1}}>1 \quad (12)$$

In our scheme, we remove each dimension and re-compute LOF value to see without which dimension of data, LOF decreases the most. For example, if the dimension is *"%memused"*, it is the most likely that memory leakage is the main reason for the abnormal status. Dimension reduction is a by-product of DR-LOF: only one dimension of data will be the input to SAX sub-system.

D. SAX Implementation

SAX works as a validation process after LOF. As mentioned before, LOF neglects the time information in data. In the context of VM live migration, the same application working on different hosts shows different characteristics. LOF cannot adapt to such changes and it will mistake normal behaviors for anomalous and vice versa.

SAX reads in the dimension of data which is the most anomalous. After symbolizing the data, it compares the distance before and after migration. If the symbolic distance between them is less than threshold, the anomalies reported by DR-LOF after migration will not be reckoned as real anomalies and the system will not raise alarms. On the contrary, if the symbolic distance is large enough, DR-LOF's alarms along with anomalies' reasons will be reported to the supervisors. Threshold can be determined by the variance of several subsequences in certain testing environment.

V. Performance Evaluation

A. Experimental Setup

Our infrastructure consists of 2 Dell R410 Rack servers (named as R4100 and R4101) powered by Intel Xeon CPU E5606 @2.13GHz. Xen virtual machines (VMs) are running on the servers with CentOS Linux operating systems. We used Xen hypervisor as the virtualization tool.

As in Fig. 4, ScaLapack [16] parallel computation programs are deployed on the VMs. This application performs large scale matrix decomposition calculation, which causes large CPU and disk consumption. Four VMs (VM008 to VM011) on Server R4101 undertake this task. VM012 works as the dispatcher to control working balance among those 4 VMs.

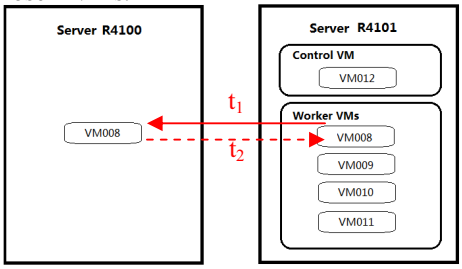

Figure 4. Enviromental relationship of servers and virtual machines.

B. Evaluation of DR-LOF

With SAR, we sampled one 38-dimensional data set every minute from 00:00:00 to 05:00:00. Three different types of anomaly were injected to see if DR-LOF can successfully recognize the type of anomalies: (1) CPU emulation at 00:00:00; (2) Routine maintenance at 04:02:00; and (3) memory leakage at 05:00:00.

Fig. 5 shows $LOF_k^n(A)/LOF_k(A)$ from 00:00:00 to 00:04:00, 04:00:00 to 04:07:00 and 05:00:00 to 05:31:00. We did not show all the results and only 7 most important dimensions are represented in the figure for legibility.

The dimension of data which new LOF value achieves the minimal is concluded in the second column in Table I of each period respectively. By comparing them with the third column, which shows the real cause of anomalies, we can see that DR-LOF correctly shows which dimension takes the most responsibility for the anomalous situation. All the dimensions with minimal LOF ratio are relevant to the type of anomaly injected. Besides, routine maintenance is also seen as an anomaly by LOF, and we can find out that in this routine maintenance, the system performs a great quantity of I/O operations firstly and then buffer transmission.

C. Anomaly Detection in VM Migration Scenarios

To demonstrate that our proposed scheme can distinguish anomalies but will not mistake normal data on new hosts after VM live migration, we judge the behavior of the scheme when the VM works normally and it is attacked by malicious port scanning.

As shown in Fig. 4, VM008 is live migrated to R4100 at time point t_1. After working on R4101 for a period of time, it is migrated back to R4101 at time point t_2.

1) VM works properly on new host (t_1=17:00:00; t_2=18:00:00)

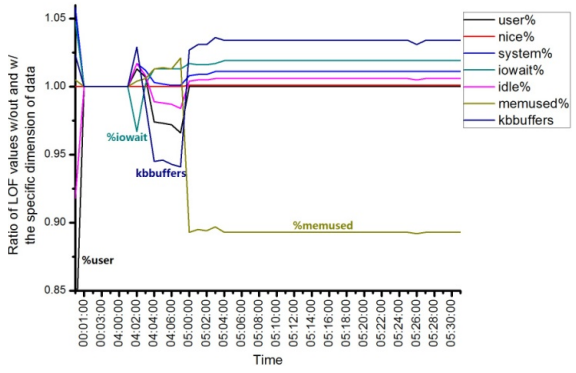

Figure 5. Ratio of LOF values with/without specific dimensions of data and all dimensions of data in the experiment to evaluate DR-LOF.

TABLE I. Result of DR-LOF and Real Anomaly Injected

Time period	Attribute with minimal LOF ratio	Anomaly injected
00:00:00	%user	CPU emulation
00:00:01-00:04:00 **04:00:00-04:01:00**	None	None
04:02:00-04:07:00	%iowait; kbbuffers	Routine maintenance
05:00:00-05:31:00	%memused	Memory leakage

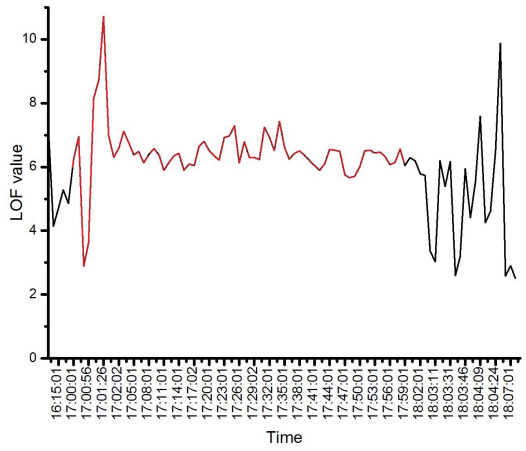

Figure 6. Anomalous LOF values with threshold greater than 2.5 in experiment 1.

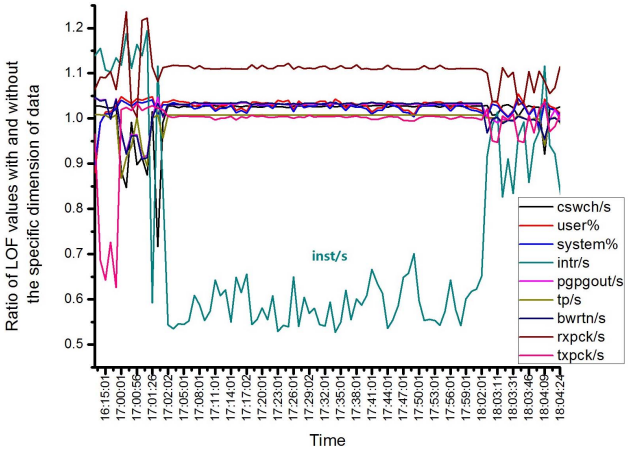

Figure 7. Ratio of LOF values without specific dimensions of data and all dimensions of data in experiment 1.

TABLE II. SYMBOLIC DISTANCE OF SEGEMENTS IN EXPERIMENT 1

Segments	Symbolic Distance
(1) and (2)	4.5057
(2) and (3)	4.3429
(1) and (3)	4.3138

During 17:00:00 to 17:59:59, VM008 works on the destination server (R4100). No anomaly is injected during this period. We implement this experiment to test whether our scheme can successfully adapt to the new infrastructures and do not regard the performance on new hosts as anomaly. Testing data is collected from 16:00:00 to 18:59:00.

First, as a comparison, LOF is implemented. Fig. 6 shows only the anomalous points (LOF > 2.5). Note that the threshold, 2.5, is an value experimentally selected through a two-day test in the same environment, which can best distinguish anomalies from normal data. The figure illustrates that most of the anomalies detected by LOF occur between 17:00:00 to 18:00:00 (the red part), which is just the time when VM008 is working on the new host. This phenomenon implies that classic LOF in [3] cannot adapt to the new host caused by VM live migration.

Fig. 7 shows the result of DR-LOF. *'intr/s'* (number of interrupt per second) is the dimension reflects the anomaly detected by LOF the best on the new host as it has the minimal new LOF value. So, we implement SAX with *'intr/s'* and compute the distance between any two of the three periods: (1) 16:00:00 to 16:59:00, (2) 17:00:00 to 17:59:00 and (3) 18:00:00 to 18:59:00, which represent the periods before, during and after VM is migrated to a new server and comes back. SAX results are shown in Table Ⅱ.

We did some experiments and selected 0.25 as the threshold of sequence matching. The variance in this test is 7.13×10^{-3}, hence, we can conclude that no anomaly happens on the new server and refuse to report the anomalies reported by LOF during period (2).

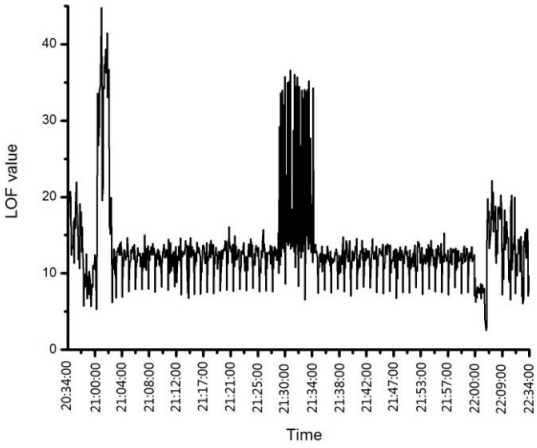

Figure 8. Anomalous LOF values with threshold greater than 2.5 in experiment 2.

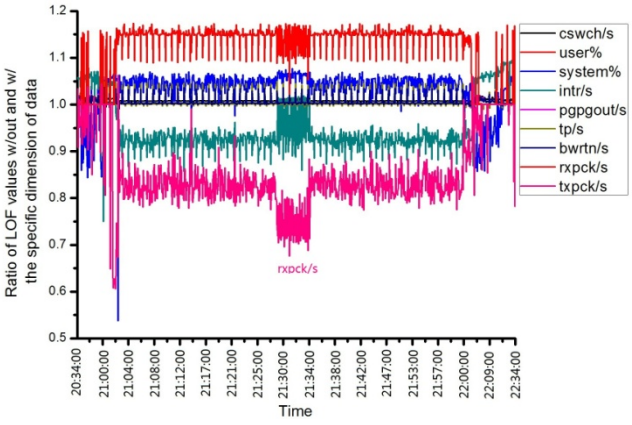

Figure 9. Ratio of LOF values without specific dimensions of data and all dimensions of data in experiment 2.

TABLE III. SYMBOLIC DISTANCE OF SEGEMENTS IN EXPERIMENT 2

Segments	Symbolic Distance
(1) and (2)	8.0383
(2) and (3)	8.1466
(1) and (3)	4.5757

2) Malicious port scanning on new host (t_1=21:00:00; t_2=22:00:00)

Similarly, VM008 is working on R4100 from 21:00:00 to 21:59:59, and gets back to R4101 at 22:00:00. During the time from 21:30:00 to 21:35:00, malicious port scanning is performed. This experiment is designed to judge whether our scheme can figure out anomalies on new hosts.

First, also served as a comparison, LOF is implemented. Note that only points with LOF > 2.5 are drawn on Fig. 8.

Then, DR-LOF and SAX are implemented. Fig. 9 shows *'rxpck/s'* (number of package sent per second) is the most significant dimension. The symbolic distances among any two of the periods: (1)20:00:00 to 20:59:00 (2) 21:00:00 to 21:59:00 and (3) 22:00:00 to 22:59:00 are shown in Table Ⅲ, variance of those data is 8.251, so, the anomalies reported by LOF in period (2) should be regarded as real anomalies.

D. Comparison with Other Anomaly Detection Methods

1) Our scheme vs. classic LOF

The differences between our scheme and classic LOF [3] lie in: (1) we introduce adaptive LOF by updating R-tree; (2) we propose DR-LOF which can identify the reason of anomalies; (3) SAX algorithm is implemented to make the scheme adaptive to VM migration and other infrastructure changes. Fig. 6 and 8 show anomalies detected by classic LOF. Detection rates and false alarm rates are shown in Table Ⅳ. Note that Experiment 1 does not have Detection Rate since no anomalies are injected. But both algorithms mistake some normal behaved data points as anomalies, so it has False Alarm Rate.

Since the experiments involve VM migration, it will confuse classic LOF by the different scale of statistics. Classic LOF has relatively high false alarm rate because it takes a long time to adapt to the new host every time VM is migrated and during that time, the results are most likely wrong. And if anomaly happens right after VM migration, classic LOF may fail to detect it.

2) Our scheme vs. Clustering

Clustering is one of the most popular anomaly detection methods. We directly implement the EM (Expectation Mean) clustering tool provided by WEKA (Waikato Environment for Knowledge Analysis) [17] with the same groups of data in Sub-section *C*. We have tested other clustering methods of other categories like partitioning (e.g. k-means, hierarchical and density-based clustering) [18] in WEKA, they generate similar results because they share the same basic mechanism. Clustering results are shown in Fig. 10 and 11 respectively.

The cluster with the minimal percentage of points is the anomaly one, cluster 2 and cluster 0 will be considered as the anomalous cluster in experiment 1 and 2 respectively. Detection rates and false alarm rates of our scheme and clustering are shown in Table Ⅴ.

TABLE IV. COMPARISON OF OUR SCHEME AND CLASSIC LOF

	Experiment 1		Experiment 2	
	Our scheme	Classic LOF	Our scheme	Classic LOF
Detection Rate	/	/	98.3%	96.7%
False Alarm Rate	0.3%	18.4%	0.9%	16.9%

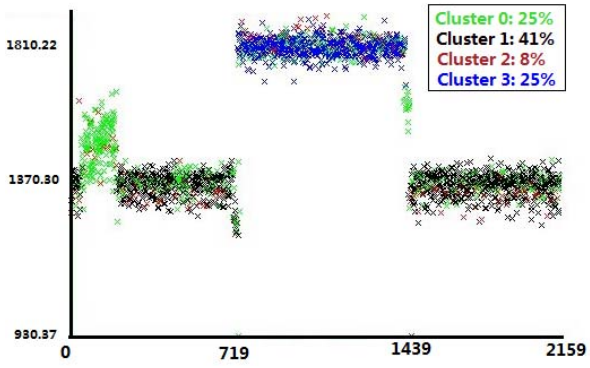

Figure 10. Clustering results in experiment 1.

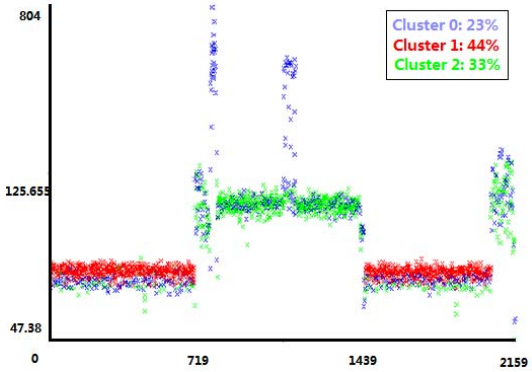

Figure 11. Clustering results in experiment 2.

TABLE V. COMPARISON OF OUR SCHEME AND CLUSTERING

	Experiment 1		Experiment 2	
	Our scheme	Clustering	Our scheme	Clustering
Detection Rate	/	/	98.3%	66.1%
False Alarm Rate	0.3%	7.8%	0.9%	21.3%

Although clustering can generally divide data into correct clusters, it cannot tell which cluster is the anomalous one in VM migration scenario. In both experiments, detection rate is extremely low because clustering mistakes cluster 2, which is normal, for anomalies. In addition, clustering mixes all the dimensions of data, which makes it impossible to detect the reason of anomalies. To conclude, clustering is not appropriate for anomaly detection in VM migration.

Supervised methods, like random forest [19], may perform better than clustering in VM migration scenarios. However, as the supervised methods require time-consuming manual interventions which are difficult in practices of on-line checking, they cannot fit into our scenarios of cloud computing.

VI. SUMMARY

In this paper, we proposed an automatic intelligent scheme to improve the security in cloud computing systems. Our scheme consists of adaptive DR-LOF and SAX. It not only adapts to new host with different infrastructures on the

context of VM live migration, but also gives possible sources that may cause the anomalies.

To identify possible sources of anomalies, we proposed adaptive DR-LOF in our scheme by extending classic LOF algorithm, which improves the performance of LOF in anomaly detection and shows the possible sources of anomalies. In experiments, DR-LOF successfully identified the anomalies of CPU emulation, memory leakage, malicious port scan and other anomalies. As far as we know, no other published work has achieved this before.

To reduce false alarm rates, we further integrated SAX in our anomaly detection scheme. Our scheme can judge the security status of the destination host after migration. Experimental results show that in the scenario of VM migration, our scheme achieves over 98% detection rate with false alarm rate under 1%, which is superior to classic LOF and clustering method.

ACKNOWLEDGMENT

This paper is sponsored in part by the Shanghai International Science and Technology Collaboration Program under Grant 13430710400, and Campus for Research Excellence and Technological Enterprise (CREATE) program of Singapore National Research Foundation under the joint project on Energy and Environmental Sustainability Solutions for Megacities (R-706-000-101-281) by Shanghai Jiao Tong University (SJTU) and National University of Singapore (NUS).

REFERENCES

[1] J. Heiser, "What you need to know about cloud computing security and compliance," online: www.gartner.com/id=1071415, 2009

[2] V. Chandola, A. Banerjee, V. Kumar, "Anomaly Detection: A Survey", ACM Computing Surveys, 2009, vol: 41, Issue: 3, pp: 1-58.

[3] M. M. Breunig, H. Kriegel, R. T. Ng, J. Sander, "LOF: identifying density-based local outliers". In Proceeding of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD '00), Jun. 2000, vol: 29, issue 2, pp: 93-104.

[4] J. Lin, E. Keoghm, S. Lonardi, B. Chiu, "A Symbolic Representation of Time Series, with Implications for Streaming Algorithms", In Proceeding of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 2003, pp:2-11.

[5] P. Kumar, N. Nitin, V. Sehgal, K. Shah, et al., "A Novel Approach for Security in Cloud Computing using Hidden Markov Model and Clustering", In Proceeding of the World Congress on Information and Communication Technologies (WICT), Dec. 2011, pp. 810-815.

[6] Z. Xie, T. Quirino, M. Shyu, "UNPCC: A Novel Unsupervised Classification Scheme for Network Intrusion Detection", In Proceeding of the 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06), 2006.

[7] W. Wang, Y. Zhang, B. Lin, X. Wu, K. Miao, "Secured and Reliable VM Migration in Personal Cloud", In Proceeding of the 2nd International Conference on Computer Engineering and Technology (ICCET), Apr. 2010, vol.1, pp:705-709.

[8] X. Chen, H. Wan, S. Wang, X. Long, "Seamless Virtual Machine Live Migration on Network Security Enhanced Hypervisor", In Proceeding of the 2nd IEEE International Conference on Broadband Network & Multimedia Technology, Oct. 2009, pp:847-853.

[9] M. Alshawabkeh, J.A. Aslam, D. Kaeli, J. Dy, "A Novel Feature Selection for Intrusion Detection in Virtual Machine Environments", In Proceeding of the 23rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI'2011), 2011, pp: 879-881.

[10] T. Huang, Y. Zhu, Q. Zhang, Y. Zhu, et al., "An LOF-based Adaptive Anomaly Detection Scheme for Cloud Computing", In Proceeding of the IEEE 37th Annual Computer Software and Application Conference Workshops, Jul. 2013.

[11] L, Lazarevic, V.K. Ertoz, A. Ozgur, J. Srivastava, "A comparative study of anomaly detection schemes in network intrusion detection". In Proceedings of the SIAM International Conference on Data Mining, 2003, pp: 25-36.

[12] N. Roussopoulos, S. Kelley, F. Vincent, "Nearest neighbor queries". In Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data, 1995, pp: 71-79.

[13] A. Guttman, "R-trees: a dynamic index structure for spatial searching". In Proceedings of the 1984 ACM SIGMOD international conference on Management of data. 1985, vol. 14, issue 2, pp: 47-57.

[14] Hwang, S., Kwon, K., Cha, S., & Lee, B. "Performance evaluation of main-memory R-tree variants". In Proceeding of the 8th International Symposium (SSTD'03), July 2003, pp:10-27.

[15] E. Keogh, J. Lin, A. Fu. "HOT SAX: Finding the Most Unusual Time Series Subsequence: Algorithms and Applications", In Proceeding of the 5th IEEE International Conference on Data Mining, Nov. 2005.

[16] Choi J., J. J. Dongarra, R. Pozo, and D. W. Walker, "Scalapack: A Scalable Linear Algebra Library for Distributed Memory Concurrent Computers", Proceedings of the Fourth Symposium on the Frontiers of Massively Parallel Computation, 1992, pp. 120-127.

[17] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. Witten, "The WEKA Data Mining Software: An update", ACM SIGKDD Explorations Newsletter, Jun. 2009, vol:11, issue:1, pp:10-18.

[18] I. H. Witten and E. Frank, "Data Mining: Practical machine learning tools and techniques". Morgan Kaufmann, 2005.

[19] L. Breiman, "Random Forests", Machine Learning, 2001, Kluwer Academic Publishers, vol: 45, pp: 5-32.