A Supervised Learning Model for Identifying Inactive VMs in Private Cloud Data Centers

In Kee Kim
University of Virginia
Charlottesville, VA, USA
ik2sb@virginia.edu

Sai Zeng
IBM T.J. Watson Research
Yorktown Heights NY, USA
saizeng@us.ibm.com

Christopher Young
IBM T.J. Watson Research
Yorktown Heights NY, USA
ccyoung@us.ibm.com

Jinho Hwang
IBM T.J. Watson Research
Yorktown Heights NY, USA
jinho@us.ibm.com

Marty Humphrey
University of Virginia
Charlottesville, VA, USA
humphrey@cs.virginia.edu

ABSTRACT

A private cloud has become an essential computing infrastructure for many enterprises. However, according to a recent study, 30% of VMs in data centers are not being used for any productive work. These "inactive" (or "zombie") VMs can arise from faulty VM management code within the cloud infrastructure but are usually the result of human neglect. Inactive VMs can hurt the performance of productive VMs, can distort internal cost management, and in the extreme can result in the cloud infrastructure being unable to allocate resources for new VMs. Correctly assessing the productivity of a VM can be challenging: e.g., is a VM that has low CPU utilization being used to slowly edit source code or is it an inactive VM that happens to be performing routine maintenance (e.g., virus-scan and software updates)? To address this problem, we develop a supervised learning model that leverages primitive information (e.g., running process, login history, network connections) of VMs periodically collected by a lightweight data collection framework. This model employs a linear support vector machine (SVM) approach that reflects single VM behavior as well as coordinated VM behaviors. We evaluated the identification accuracy of this model with a real-world dataset within IBM of more than 750 VMs. Results show that our model has a 20% higher accuracy (90%) than state-of-the-art approaches. An accurate model is an important first step to enable private cloud infrastructures to achieve better resource management through such actions as suspending or dynamically downsizing inactive VMs.

CCS Concepts

•Information systems → Data centers; •Networks → Cloud computing; Data center networks; •Software and its engineering → Cloud computing;

Middleware Industry '16, December 12-16, 2016, Trento, Italy

DOI: http://dx.doi.org/10.1145/3007646.3007654

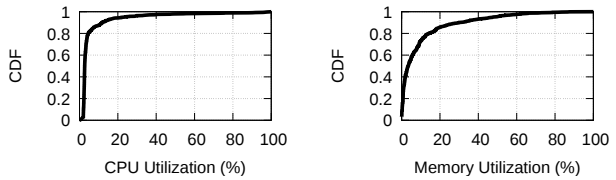
Keywords

Cloud Computing, Identifying Inactive VMs, Data Center Management, IaaS Garbage Collection, Supervised Learning

1. INTRODUCTION

Over the last decade, there has been a huge transition from traditional in-house computing infrastructure to private clouds [18]. Clearly, private clouds offer many benefits as compared to traditional IT infrastructure. e.g., fine-grained resource use via virtualization, pay-as-you-go pricing model, simplified process to obtain and release VM (Virtual Machine) instances [1].

However, one of the benefits – *the simplified process to obtain and release VMs* – can also lead to problems. In private clouds, VM execution is frequently either "free" or of marginal "cost" (e.g., internal billing across departments within an organization never directly makes it to individual employees). As such, there can be little direct incentive for judicial management of one's running VMs, which of course leads to VMs persisting that are of little value. According to a recent study [14], more than 30% of VMs in enterprise data centers are "comatose", meaning that these VMs are not being used for any productive work in their organizations. These "inactive" (or "zombie") VMs waste a huge amount of capital investment for maintaining the infrastructures and reduce available capacity of private clouds. Inactive VMs can hurt the performance of productive VMs, can distort internal cost management, and in the extreme can result in the cloud infrastructure being unable to allocate resources for new VMs.



(a) CPU Utilization (b) Memory Utilization

Figure 1: CDF of VM Utilization in an Enterprise Data Center (measured with 200 VMs in total).

Intuitively, low resource utilization within the VM might be sufficient to identify inactive VMs. However, Figure 1

shows the results from when we measured the CPU and memory utilization of 200 VMs in an enterprise data center for 5 business days. Figure 1 shows that most VMs have low resource utilization. i.e., 80% of VMs use less than 10% of CPU and 20% of memory resource. While it is certainly possible that all of these VMs are inactive, our subsequent ad hoc investigations on a per-VM basis indicate they are not. Similarly, we have observed times where inactive VMs look very active due to high resource consumption. e.g., automatic software updates and virus-scans can consume high CPU and memory resources for a short period of time.

In this work, we create a supervised learning model that accurately identifies active VMs in data centers. This model leverages a range of primitive information of VMs – *VM utilization, running processes, login history, network connections, I/O usage, and others* – periodically collected by a lightweight data collection framework. In a public cloud setting, measurement mechanisms via such OS modifications would generally be prohibited because of privacy concerns [11, 19, 2]. However, in this work, we leverage the nature of the private cloud environment, in which such modifications are supported and generally considered to be in-line with enterprise goals as a whole, to measure and collect such information.

This model infers the purpose of a VM via its running processes and subsequently selects the most important features for the active-or-inactive classification according to the purpose of VMs. Direct user feedback with random sampling is used to find specific features that are highly correlated with identifying active/inactive VMs with different purposes. This model employs a linear SVM (Support Vector Machine) for identifying active/inactive VMs with the specific features. To minimize misclassification error, this model also analyzes network dependencies among VMs and adjusts identification results.

To evaluate the identification accuracy of our model, we collect real-world data of more than 750 VMs in IBM data centers and compare the identification accuracy with ground truth from real VM users from the IBM research division. The experiment results show that our model has a 20% higher accuracy than other state-of-the-art approaches.

The contributions of this paper are:

- We have created a simplified data discovery and collection framework for active/inactive VM identification.
- We have collected user feedbacks associated with identifying active and inactive VMs and used the feedbacks to design a method for identification.
- We have created a model to accurately identify active VMs in private cloud data centers.
- We have validated the model with real-world large scale dataset comparing with state-of-the-art approaches.

The rest of this paper is organized as follows: Section 2 describes design of the identification model in detail. Section 3 provides evaluation setup and results. Section 4 contains related work. Section 5 concludes this paper.

2. DESIGN OF IDENTIFICATION MODEL

This section describes two main parts of identifying active/inactive VMs: 1) a lightweight data collection/discovery framework and 2) an identification model. The data collection framework periodically crawls usage information of target VMs and identification model determines active/inactive VMs from the collected dataset. Figure 2 shows system architecture that collects VM data and identifies them as active or inactive.

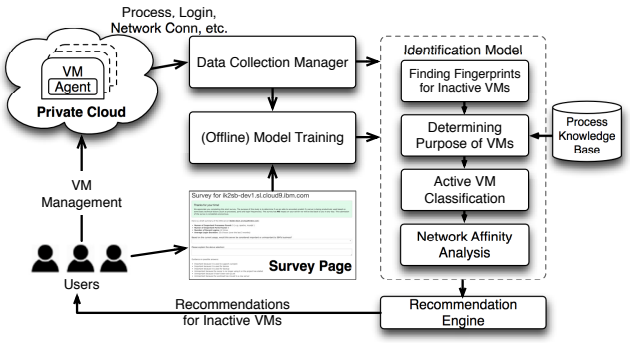


Figure 2: Active/Inactive VM Identification Framework.

2.1 Data Collection

We begin with design and implementation of the data collection framework, which consists of agents and a collection manager (the left side of Figure 2). The agents are running on target VMs and execute a bash shell script in every predefined time interval (e.g., in every four hours). The script gathers primitive, but useful following information by using basic commands supported by operating systems (e.g., ps, netstat).

1. **Host Information**: host name and IP addresses of target VMs, and timestamp for data collection.
2. **Resource Usage**: CPU, memory, and I/O utilization of target VMs.
3. **Process Information**: a list of all running processes, and resource utilization of each process.
4. **Network Connections**: all open ports from processes and established connections with external entities.
5. **Login History**: login frequency/duration of users, differentiated login frequency/duration for daytime and nighttime.

The size of gathered data is less than 50 Kbytes for each VM. The collected information is transferred to the data collection manager using curl[1] and the data collection manager stores the dataset used by our identification model. Although, we rely on an IBM proprietary infrastructure management system in order to quickly implement and deploy this framework to data centers, but this system can be simply replaced by other tools like chef[2] or puppet[3]. We deploy this framework to multiple data centers after validating the safety of this framework because it should not mess up such production infrastructures.

[1]https://curl.haxx.se/
[2]https://www.chef.io/
[3]https: //puppet.com/

2.2 Data Analysis

With the dataset, the next step is to find which factors are highly correlated with activeness and inactiveness of VMs. We choose dataset from the smallest data center with 70 VMs[4] and perform correlation analysis of various factors in order to find commonly correlated features with active/inactive VMs. We ask actual users of these 70 VMs to identify the VMs and use the user feedback as ground truth. We consider a wide range of factors[5] from data set and calculate Pearson correlation coefficient [22] of each factor with equation-(1). If a factor's coefficient is close to 1.0, the factor is highly correlated with a VM's identification. However, in this analysis, we cannot find any significant factors and all factors have $0 < |\rho| < 0.5$.

$$\rho = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}} \quad (1)$$

We discuss with the VM users with respect to this results (no common correlations) and ask them actual reasons why the VMs are inactive or active. With this discussion, we realize that VMs with different purposes have different usage pattern to determine inactive or active VMs. For example, some VMs used for big-data analytics highly utilize CPU and memory resources, whereas other VMs used for development show low resource utilization but have very frequent user access mostly at daytime. Also, many active VMs are temporarily inactive, meaning that the users keep the VMs for future use or many VMs contain long-lived and mostly idle applications (e.g., mostly lightweight web application) [23]. This observation means that leveraging such single dimension information is not a sufficient approach to accurately identify VM's activeness or inactiveness.

With this observation, we manually label purpose of each VM and re-calculate correlation coefficient of the diverse factors. Table 1 shows correlation coefficient according to the purpose of VMs. Due to page limitation, we only show correlations for two use cases; analysis and development purpose. As shown in Table 1, we can successfully find strong correlations for identification. We extract specific correlated features for five particular purposes of VMs with manual labeling. These features will be used by our identification model described in the next section. However, these features are useful only if purposes of VMs are known. It is still challenging to properly determine the purpose of each VM. We will discuss more about how to determine the purpose of each VM in the next section.

2.3 Identification Model

We design a supervised learning-based identification model with the data analysis results. This model uses the dataset crawled by the data collection framework and determines active and inactive VMs through the following four pipeline steps (the right side of Figure 2):

[4]These VMs are excluded in our evaluation for model accuracy in Section 3.

[5]We extract and use total 27 factors from collected dataset including CPU/memory utilization of both VMs and significant processes, logic frequency/duration, daytime/nighttime logins, connections for significant process, external/local connections, etc.

[5]User development activity process is one of 25 classes for significant user applications and includes editing (e.g., vi), compile-related, and git-related activities.

Table 1: Example of Correlated Features based on VM Purpose.

Purpose	Factor	ρ
Analytics	CPU% of significant user procs.	0.95
	MEM% of VM.	0.90
	# established connections.	0.97
Development	# Logins $> n$ times	0.85
	Daytime logins hours$> m$ hr.	0.91
	# User dev. activity procs[6].	0.88

1. Finding fingerprints for inactive VMs.
2. Determining purpose of target VMs.
3. Active VM classification with linear SVM.
4. Analyzing network dependencies of VMs.

This model identifies active VMs first and treats a compliment set as inactive VMs. The reasons that this model focuses more on finding active VMs are:

- Active VMs are highly likely to have more strong features to determine them as active VMs.
- Population of active VMs is larger than population of inactive VMs. Approximately 70% of VMs in data centers can be considered as active VMs [14].

Due to these reasons, it makes more sense to us to concentrate on designing mechanisms to correctly identify active VMs first. Moreover, we need to minimize false negative errors[7] since these errors have huge impact on both organizations and the VM's user. If data center operators suspend or terminate this VM based on the identification result, the actual owner of the VM could (temporarily) lose all data and application configurations in the VM and this could be a SLA (Service Level Agreement) violation case.

2.3.1 *Finding Fingerprints for Inactive VMs*

This step is to filter inactive VMs out from the dataset using four simple, yet effective rules. If a VM satisfies these four rules, it is considered as an inactive VM. The rules are:

- **Rule #1** – No reboot of a VM over last 6 months.
- **Rule #2** – No significant[8] processes on VMs.
- **Rule #3** – No login activities over last 3 months.
- **Rule #4** – No established connections with other external entities during data collection period.

If a VM satisfies these four rules, this implies that this VM is highly likely inactive since it has no maintenance processes (rule #1 – no reboot over last 6 months), very unclear purposes (rule #2 – no interesting/significant processes), and no activities from any external entities (rule #3 – no user logins and rule #4 – no applications connections). With these rules, our model excludes VMs from the identification target.

2.3.2 *Determining the Purpose of Target VMs*

This step is to apply our important findings (Section 2.2) for the identification process. The model determines the purpose of each VM based on running processes. For example, if RDBMS processes (e.g., PostgreSQL, MySQL) are

[7]False negative case is "active VMs are identified as inactive."

[8]We define 25 classes of significant user applications and ignore kernel or OS related processes.

Table 2: Specific Features According to Purpose of VMs used by Active VM Classification.

Purpose of VM	List of Specific Features
Analytics	%CPU/%MEM of VM, %CPU of significant user procs, # of open ports, # of established conns.
DevOps	# of significant user procs, %CPU/%MEM of significant user procs, # of established conns.
Development	# of logins, average login hours (daytime), # of ssh/VNC conns. # user dev. activity procs.
Storage/Backup	# of storage/backup procs, I/O usage, %MEM of significant user procs, # of established conns.
Others	# of user maintenance activity procs, # of daytime logins.

running on a VM, this VM is supposed to be mainly used for storage purpose. Similarly, if hadoop processes are running on a VM, this VM is highly likely to be used for analytics purpose. However, this insight does not necessarily work for all cases. RDBMS is mainly used to store structured data, but it does not always mean that this VM is only used for storage-purpose. In many cases, RDBMS can be part of an application configuration (e.g., LAMP[9]) for development-purpose or test-purpose VMs.

To properly decide possible purposes of VMs, we leverage user feedbacks for 70 VMs (used for data analysis in Section 2.2) and decide weights for purposes associated with running processes. We then create a function that maps type of running processes to possible purposes. This function returns multiple purposes with different weights. (e.g., **input**: $process_i \rightarrow$ **output**: $\{purpose_1 : w_1, purpose_2 : w_2, ..., purpose_n : w_n\}$). The weights associated with different purposes of VMs are directly used by linear SVM classifier in the next step.

2.3.3 Active VM Classification with Linear SVM

The identification model employs a supervised learning model called a linear SVM in order to classify active VMs. SVM is a widely used classifier to solve diverse research problems in cloud computing area (e.g., workload prediction/classification [13, 5], application performance modeling [15], and anomaly detection [10]). Linear SVM is an optimal margin-based classifier with linear kernel and tries to find a small number of data points that separate all data points of two classes with a hyperplane [9].

A key insight using this supervised learning approach is that the linear SVM is dynamically leveraging different correlation features for the identification according to the purposes of target VMs. Table 2 describes specific correlated features, with respect to possible purposes of target VMs, dynamically used by SVM classifier. As shown in Table 2, both "analytics"- and "devops[10]"-purpose mostly use VM utilization-related features, "development"- purpose selects user access and development related processes (e.g., git[11]) as features, and "backup/storage"-purpose focuses on particular utilization features associated with storage/backup operations (e.g., I/O, memory) for SVM classification. "Others"-purpose is the most challenging case in this classification process because most of these VMs do not have any interesting features in running processes or utilization. So the SVM classifier can use only two features; daytime login frequency and maintenance-related features (e.g., yum, apt-get).

Moreover, VMs can be used for multiple purposes, so the identification model runs the SVM classifier multiple times with different weights. The VMs then are classified as active or inactive with equation-(2) where w is a weight for a particular purpose for the classification and ψ is the classification results with the particular purpose ($\psi \in \{0, 1\}$).

$$Classification\ Result = \frac{\sum_{i=1}^{n} w_i \times \psi_i}{\sum_{i=1}^{n} w_i} \quad (2)$$

2.3.4 Analyzing Network Dependencies of VMs

The last step of the identification model is to analyze network dependencies of all VMs and validate the classification result with the network dependencies. This step is designed to minimize false negative errors with network affinity analysis [12, 4, 17], which is a well-known approach for enterprise application migration to cloud data centers. The identification model investigates all external connections of VMs that are classified as inactive. This model adjusts the VM's classification result from inactive to active if there is a network dependency with active VMs. This step is particularly useful for VMs with cluster configurations (e.g., hadoop[12] or mesos[13]). Usually, a master VM of such clusters has strong features to be properly classified as active or inactive VM, but slave VMs have weak features for the identification process. Thus, network affinity analysis propagates confidence determined from a master VM to slave ones. In the evaluation section, we will assess performance of the network affinity analysis on how much it can contribute to accuracy improvement for our identification model.

3. MODEL EVALUATION

3.1 Evaluation Setup

We describe experiment setup to evaluate our identification model.

Dataset: We deploy the lightweight data collection framework to multiple data centers for IBM private cloud infrastructure and collect primitive data – *utilization, process, login history, network connections, etc.* – from 750 VMs[14] in every four hours. Total data collection period is four weeks. To obtain ground truth for dataset, we create a survey system and ask actual users to identify that their VMs are active or inactive.

Evaluation Metrics: We evaluate accuracy of our identification model in isolation and measure the impact of network affinity analysis on improving the accuracy of our model. We then compare the accuracy of the identification model with two state-of-the-art approaches. For both evaluations, we validate the accuracy with k-fold cross validation [9].

Accuracy Metrics: We use three accuracy metrics; recall, precision, and f-measure [9]. Recall ($\frac{TP}{TP+FN}$) is more sensitive to false negative errors and precision $\frac{TP}{TP+FP}$ is more sensitive to false positive errors. F-measure is calculated by

[9]Linux, Apache Web Server, MySQL, and PHP

[10]A clipped compound of development and operation [21].

[11]https://git-scm.com/

[12]http://hadoop.apache.org/

[13]http://mesos.apache.org/

[14]The total number of owners for the 750 VMs is about 200.

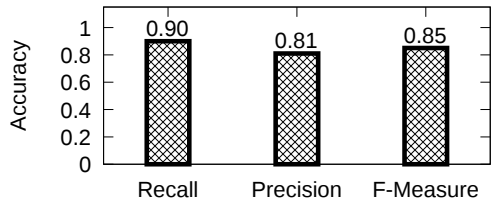


Figure 3: Accuracy of Our Identification Model.

Table 3: Impact of Network Affinity Analysis (NAA).

Metric	Recall	Precision	F-Measure
Without NAA	0.83	0.78	0.82
With NAA	**0.90**	**0.81**	**0.85**
Improvement	7%	3%	3%

$\frac{2 \times Recall \times Precision}{Recall + Precision}$ and is a harmonic mean of recall and precision. (TP: True Positive, TN: True Negative, FP: False Positive, and FN: False Negative.)

False negative errors in this evaluation is a situation where an active VMs are not correctly identified as active. These false negative errors are seriously concerned by this work since these type-II errors (false negatives) have huge impact on both enterprise and VM users, so among these three accuracy metrics, recall is the most important for this evaluation.

Baselines: We use two state-of-the-art approaches as baselines; Pleco [20] and Garbo [6]. Pleco identifies active and inactive VMs with a combination of a reference model and decision tree classification. Garbo is a graph-based VM cleaning-up tool for Amazon Web Services[15]. Garbo creates a directly acyclic graph from core nodes and performs *mark* and *swap* operation to clean up inactive VMs.

3.2 VM Identification Accuracy

We evaluate accuracy of our identification model in isolation and use dataset/ground truth of 750 VMs. The results are shown in Figure 3. Our model can identify active VMs with 0.90 of recall, 0.81 of precision, and 0.85 of f-measure. In this evaluation, the model identifies 472 (63%) of active VMs and 278 (37%) of inactive VMs. Especially for the recall, this model has 47 false negative errors. The inactive VM rate (37%) is slightly higher than the observation result (30%) from the recent study [14]. This might be because we collect VM data from private clouds for research and development division and characteristics of VMs could be little different from production clouds.

To evaluate the impact from network affinity analysis (Section 2.3.4), we measure the accuracy of our model with and without the network affinity analysis step. The evaluation results are shown in Table 3. Network affinity analysis can improve accuracy of our model with 3% – 7% and has more impact on increasing recall (7%). In order to understand this impact, we measure the number of external connections from all 750 VMs. We found that active VMs have average of 2.3 external connections ($\sigma = 4.1$) and inactive ones are connected with 0.5 of external connections in average ($\sigma = 0.9$). This observation indicates that active VMs are highly likely to have connections with other active VMs.

[15]https://aws.amazon.com/

Table 4: Comparison of Technical Differences.

Approach	Details
Our Model	Primitive Information (e.g., utilization, running processes, login history, etc.) + SVM Classifier + Network Affinity Analysis.
Pleco	Reference Model + Decision Tree Classifier.
Garbo	Graph-based Dependency Discovery.

3.3 Comparison with Baselines

The next evaluation step is to compare the accuracy of our identification model with two baselines. For accuracy comparison, we measure recall, precision, and f-measure of three approaches. Figure 4 illustrates evaluation results of three metrics for all approaches. In all metrics, our identification model shows higher accuracy (11% – 20%) than the results from two baselines. Pleco and Garbo only have 0.75 and 0.70 of recall, but our model shows 0.90. To understand these results, we need to see the technical differences in the three approaches.

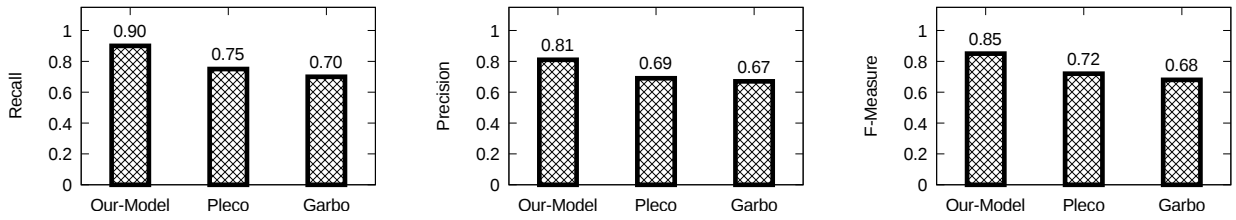
Table 4 summarizes technical differences of main mechanisms in three approaches. Two baselines – Pleco [20] and Garbo [6] – are a graph-based approach, indicating that connectivity information of among cloud resources (e.g., VM) is the most critical to identify active and inactive resources. As we confirmed in Section 3.2. connectivity (or affinity) information could be a key to determine active/inactive VMs, but other information (e.g., login, utilization) is also very critical. For example, in many cases, VMs performs significant processing without any external connections. In our measurement, 30% of active VM (142 VMs out of 472 VMs) have no connections with external entities. These (stand-alone) VMs can be very challenging use cases for the graph-based approach to correctly identify them as active or inactive resources. Since VMs are created and used for different purposes, identification models should cautiously select proper features from multiple dimensions of information including network connectivity/dependency, utilization, login history, and others.

Moreover, the graph-based approaches should properly define starting nodes and discovery scope since VMs can be connected to/from a number of external entities located inside or outside of the target data centers. If too small or too large number of starting nodes are selected, it can easily hurt accuracy of approach or have high computational overhead. Thus, defining proper discovery size/staring nodes is very challenging to maximize accuracy and minimize overhead of the graph-based approaches.

4. RELATED WORK

There are two approaches for the infrastructure garbage collection with active/inactive VM identification: graph-based [20, 6] and rule-based [16, 7, 3]. The graph-based approach leverages a network affinity model that references connectivity between VMs, and propagates importance of VMs to connected ones. The rule-based approach makes use of pre-defined rules (or annotations) to identify any VMs that violate the rules.

Graph-based approach: Pleco [20] is a tool to detect inactive VMs in IaaS clouds. Similar to memory garbage collector which identifies garbage objects by examining object references, Pleco constructs a VM reference model according to a dependency of applications (network affinity model), as-



(a) Recall (b) Precision (c) F-Measure

Figure 4: Accuracy Results of Three Approaches: Our Identification Model, Pleco, and Garbo.

signs weight to each reference, and calculates a confidence level for identified inactive VMs. Since only affinity information is used, the recall of this approach is low (75%).

Garbo [6] generates a directed graph with connectivity (dependency) of resources and adopts an idea from garbage collection mechanism. However, in many cases, stand-alone cloud resources, having no dependency with other cloud resources, cannot necessarily be identified as unused resources by only using resource dependencies. For example, stand-alone cloud resources often have very important dataset for an organization (e.g., key files) and many "development-and-test" VM instances oftentimes contain all tier components (e.g., LAMP) for services and they might not have any connection with other cloud resources.

Rule-based approach: Janitor Monkey [16] identifies unused resources based on a set of rules defined by operators. This set of rule is focused on the "age factor" of resources. For instance, Janitor Monkey marks resources inactive if they are not used for more than 3 days. Since operators cannot predict when the resources will be used, defining proper rules is a challenging task for the operators of cloud infrastructure management. As Janitor Monkey does not consider any connectivity (or dependency) among resources as well as the significance of the target resources, it cannot adapt to any dynamic scenarios.

Poncho [7] is used to maintain Magellan cloud[16] at Argonne (open-stack based). The core idea is to use annotation and notification. Resource owners are required to provide detailed annotations (their own rules) for their resources. The rules are not meant for global uses, but defined for specific workloads. The examples of annotations include *reboot_when*, *terminate_when*, or *snapshot_on_terminate*. *terminate_when* means that the VM can be terminated after X hour execution, and this implicitly offers VM to be used for another job. With pre-defined annotations by resource owners, inactive time of cloud resources can be minimized as well as resource utilization of the entire infrastructure can be improved, but each time users create VMs, they have to annotate VMs and the annotations need to be changed when the purpose of VMs is changed.

CloudVMI [3] offers VMI (commonly supported by Hypervisor) capability to public cloud users. This CloudVMI has a "garbage collection" functionality, which destroys VMs based on their access history. CloudVMI periodically checks "access history" of VMs since they were last used. If VM instances have not been accessed for a specific time, the VMs are destroyed by the garbage collector. This CloudVMI only considers the access history, but does not leverage resource utilization, significance, or dependency (connectivity).

[16]http://www.alcf.anl.gov/magellan

The graph-based and rule-based approaches are limited to achieve efficient active/inactive VM identification. To improve the accuracy and dynamicity, our approach designs a model that considers more aspects of systems such as significance (role) and utilization of target resources as well as connectivity among resources in our discovery scope.

5. CONCLUSION AND FUTURE WORK

We have described a supervised learning model to accurately identify active and inactive VMs for an enterprise private cloud. In order to improve the accuracy, we propose an identification model with four steps. The identification model finds fingerprints for inactive VMs first, then determines the purpose of each VM. By leveraging the supervised learning technique (SVM), we identify active VMs with corresponding features to the purpose. Finally, we validate the identification is correct using a network affinity model, and propagate importance to connected VMs. We have demonstrated how we increase the recall of the model. Our evaluation with 750 VMs from enterprise data centers shows the accuracy is 90%, and this is 15% – 20% higher than existing methods. We are closely looking at how we remove the false negatives which can have significant impact on recommendations to VM users. With the identification results, the current recommendation to VM users is straightforward. If users' VMs are identified as inactive, they are required to take proper actions with our recommendations. i.e., terminating VMs with snapshot, resizing VMs.

This accurate identification model is an important first step to enable private cloud to achieve better resource management. In the near future, based on this model, we will investigate a novel scheduling mechanism with over-provisioning for private cloud data centers that will automatically identify inactive VMs and switch them with new or active VMs. This approach will focus on improving data centers' utilization and capacity without SLA violations.

In this work, we have seriously concerned protecting VM users' privacy since some information we crawl might violate VM users' privacy. Although we assume that there is no personal data in enterprise private cloud, data collection mechanism must carefully gather the minimal amount of data for identification and strictly follow data protection regulations such as EU's the general data protection regulations [8].

6. ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions. Our grateful thanks also go to Ivan Dell'Era, Carlos Fonseca and Dr. Nikos Anerousis at IBM for their timely and valuable supports for this project.

7. REFERENCES

[1] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. A View of Cloud Computing. *Communications of the ACM*, 53(4):50–58, Apr. 2010.

[2] Microsoft Azure. Trust Center. https://azure.microsoft.com/en-us/support/trust-center/. ONLINE.

[3] Hyun Wook Baek, Abhinav Srivastava, and Jacobus Van der Merwe. CloudVMI: Virtual Machine Introspection as a Cloud Service. In *IEEE International Conference on Cloud Engineering (IC2E '14)*, Boston, MA, USA, Mar. 2014.

[4] Kun Bai, Niyu Ge, Hani Jamjoom, Ea-Ee Jan, Lakshmi Renganarayana, and Xiaolan Zhang. What to Discover Before Migrating to the Cloud. In *IFIP/IEEE International Symposium on Integrated Network Management (IM '13)*, Ghent, Belgium, May. 2013.

[5] Ron C. Chiang, Jinho Hwang, H. Howie Huang, and Timothy Wood. Matrix: Achieving Predictable Virtual Machine Performance in the Clouds. In *11th International Conference on Autonomic Computing (ICAC '14)*, Philadelphia, PA, USA, Jun. 2014.

[6] Netanel Cohen and Anat Bremler-Barr. Garbo: Graph-based Cloud Resource Cleanup. In *2015 ACM Symposium on Cloud Computing (SoCC '15)*, Kohala Coast, Hawaii, USA, Aug. 2015.

[7] Scott Devoid, Narayan Desai, and Lorin Hochstein. Poncho: Enabling Smart Administration of Full Private Clouds. In *27th USENIX Large Installation System Administration Conference (LISA '13)*, Washington D.C., USA, Nov. 2013.

[8] European-Union. The general data protection regulation. http://www.consilium.europa.eu/en/policies/data-protection-reform/data-protection-regulation/. ONLINE.

[9] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The Element of Statistical Learning: Data Mining, Inference, and Prediction. 2011.

[10] Shin-Ying Huang and Yen-Nun Huang. Network Traffic Anomaly Detection based on Growing Hierarchical SOM. In *The 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN '13)*, Budapest, Hungary, Jun. 2013.

[11] IBM. Softlayer – Privacy Agreement. https://www.softlayer.com/sites/default/files/softlayer_privacy_agreement_-_may_2016.pdf. ONLINE.

[12] Jill Jermyn, Jinho Hwang, Kun Bai, Maja Vukovic, Nikos Anerousis, and Salvatore Stolfo. Improving Readiness for Enterprise Migration to the Cloud. In *The 15th ACM/IFIP/USENIX Middleware Conference (Middleware '14)*, Budapest, Hungary, Dec. 2014.

[13] In Kee Kim, Wei Wang, Yanjun Qi, and Marty Humphrey. Empirical Evaluation of Workload Forecasting Techniques for Predictive Cloud Resource Scaling. In *2016 IEEE International Conference on Cloud Computing (CLOUD '16)*, San Francisco, CA, USA, Jun. 2016.

[14] Jonathan Koomey and Jon Taylor. 30 percent of servers are 'comatose'. http://anthesisgroup.com/wp-content/uploads/2015/06/Case-Study_DataSupports30PercentComatoseEstimate-FINAL_06032015.pdf. ONLINE.

[15] Palden Lama and Xiaobo Zhou. AROMA: Automated Resource Allocation and Configuration of MapReduce Environment in the Cloud. In *9th ACM International Conference on Autonomic Computing (ICAC '12)*, San Jose, CA, USA, Sep. 2012.

[16] Netflix. Janitor monkey. https://github.com/Netflix/SimianArmy/wiki/Janitor-Home. ONLINE.

[17] Michael Nidd, Kun Bai, Jinho Hwang, Maja Vukovic, and Michael Tacci. Automated Business Application Discovery. In *IFIP/IEEE International Symposium on Integrated Network Management (IM '15)*, Ottawa, Canada, May. 2015.

[18] Melanie Posey Robert P. Mahowald, Chris Morris, Mark Schrutt, Satoshi Matsumoto, Vladimir Kroa, Petr Zajonc, Linus Lai Frank Gens, Hamza Naqshbandi, Melih Murat, David Senf, David Tapper, Alejandro Florean, Mayur Sahni, Thomas Dyer, and Benjamin McGrath. Worldwide Hosted Private Cloud Services Forecast, 2015–2019: New Models for Delivering Infrastructure Services. In *IDC Tech Report*, Sep. 2015.

[19] Amazon Web Services. Data Privacy. https://aws.amazon.com/compliance/data-privacy-faq/. ONLINE.

[20] Zhiming Shen, Christopher Young, Sai Zeng, Karin Murthy, and Kun Bai. Identifying Resources for Cloud Garbage Collection. In *12th International Conference on Network and Service Management (CNSM '16)*, Montreal, Quebec, Canada, Nov. 2016.

[21] Wikipedia. Devops: Development and operation. https://en.wikipedia.org/wiki/DevOps. ONLINE.

[22] Wikipedia. Pearson product-moment correlation coefficient. https://en.wikipedia.org/wiki/Pearson_product-moment_correlation_coefficient. ONLINE.

[23] Liang Zhang, James Litton, Frank Cangialosi, Theophilus Benson, Dave Levin, and Alan Mislove. Picocenter: Supporting long-lived, mostly-idle applications in cloud environments. In *2016 European Conference on Computer Systems (EuroSys '16)*, London, UK, Apr. 2016.