Towards Security-Aware Virtual Server Migration Optimization to the Cloud

Bowu Zhang
Marist College
Poughkeepsie, NY, USA
bowuzhang08@gmail.com

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

Liran Ma
Texas Christian University
Fort Worth, TX, USA
l.ma@tcu.edu

Timothy Wood
George Washington University
Washington DC, USA
timwood@gwu.edu

***Abstract*—Cloud computing, featured by shared servers and location independent services, has been widely adopted by various businesses to increase computing efficiency, and reduce operational costs. Despite significant benefits and interests, enterprises have a hard time to decide whether or not to migrate thousands of servers into the cloud because of various reasons such as lack of holistic migration (planning) tools, concerns on data security and cloud vendor lock-in. In particular, cloud security has become the major concern for decision makers, due to the nature weakness of virtualization – the fact that the cloud allows multiple users to share resources through Internet-facing interfaces can be easily taken advantage of by hackers. Therefore, setting up a secure environment for resource migration becomes the top priority for both enterprises and cloud providers. To achieve the goal of security, security policies such as firewalls and access control have been widely adopted, leading to significant cost as additional resources need to employed. In this paper, we address the challenge of the security-aware virtual server migration, and propose a migration strategy that minimizes the migration cost while promising the security needs of enterprises. We prove that the proposed security-aware cost minimization problem is NP-hard and our solution can achieve an approximate factor of 2. We perform an extensive simulation study to evaluate the performance of the proposed solution under various settings. Our simulation results demonstrate that our approach can save 53% moving cost for a single enterprise case, and 66% for multiple enterprises case comparing to a random migration strategy.**

***Keywords*—Cloud Computing; Cloud Migration; Cloud Security; Cost Minimization**

I. Introduction

In the last few years, the cloud computing has emerged as a revolutionary approach for enterprises to provide cost effective and reliable services over the Internet. There has been an increasingly upward trend that enterprises move their local servers to public/private/hybrid clouds, or at least part of their machines to the cloud (i.e., cloud bursting). This "on-premise-to-cloud" migration not only helps enterprises save capital and operational expenses, but also provides enterprises with elastic resource provisioning and ready-to-use management services. Today, users can enjoy using the cloud over a one-click shop platform with a pay-as-you-go model, which in turn tremendously promotes the enterprise migration. Moreover, the diversity that lies in cloud service offerings also attracts great attention, further blooming the usage of cloud in industries.

Despite significant interests, migrating enterprise-scale workloads to the cloud[1] has been technically challenging [18, 14, 13]. One obstacle in way of the massive transition to the cloud is the lack of holistic migration techniques. Existing tools such as Rackware [28], Racemi [27], and Double-Take [10] are not able to accommodate various requirements from customers and constraints existing in cloud servers. In particular, the concern on migration expense has long been at heart of enterprises due to the fact that the migration process is quite resource consuming because of operations such as topology adaptation, resource right-sizing, and remote management. Recent works [22, 9, 29] have studied data migration strategies extensively from the perspective of cost minimization. While the majority of these works focus on reducing cost introduced by resource assignment and topology, few have taken the security expense of data centers into account.

As more and more corporations are aware of the security threats that are emerging, the security issues have become the primary concern of enterprises when managing their machines [1]. Today, cloud users face enormous security challenges posed by data loss and data leakage due to the complex computing environment. In order to satisfy the desired level of security of enterprises, the cloud community has been calling for methods to continuously monitor system activities against distributed denial of service attacks, and to efficiently control remote access towards resources in response to rising threats of data breaches. While a growing body of research [1, 19] has been devoted to helping enterprises set up security policies for their existing services in the cloud, few has considered the cost derived from security implementations, while the deployment of security defenses such as firewall network appliances (one of the most expensive resources) can incur a large amount of resource consumption in terms of time, complexity, and labor. In addition, none of previous work in literature has included security implementations at the runtime of migration, where the operations would be easy (compared to adding security to services after the migration which may involve a large amount of work to "isolate" services in order to not affect other services in the same server) and the security cost can be reduced significantly as it would be both time and resource efficient for cloud providers to design a security-aware migration plan.

In this paper, we focus on reducing the migration cost derived from security implementations in the perspective of

[1]The migration in this paper refers to live migration or copy/sync up/cut over images so that there is minimum down time (minimum service disruption).

a cloud provider by addressing clients' security needs during their migration processes. In real world, considering the cost and the nature of services, enterprises normally choose to manually add security protection only for some sensitive services (i.e., database servers and file servers), while less sensitive components (i.e., middleboxes) are directly integrated into the cloud and operated under cloud's default security settings. Considering the fact that a cloud server/pod usually has a capacity that can support multiple services simultaneously, as a cloud provider, to reduce the migration cost due to security implementation, an intuitive solution is to group machines with security needs, and migrate them to as few cloud servers as possible. With this stepping-stone idea, in this paper, we investigate the migration problem in a situation that multiple enterprises request to move their servers, databases, applications from existing physical/virtual machines to the cloud. Specifically, we focus on satisfying the security needs of customers while minimizing the cost introduced by planned security policies in the cloud by grouping servers based on their security needs. To the best of our knowledge, this is the first work in literature that considers the security policy implementation during the migration process. In addition, this is also the first time to include the security cost while considering the migration cost. Our contributions are summarized below:

- We provide a problem formulation for the security-aware virtual server migration with an objective to minimize the migration cost, and prove the proposed problem is NP-hard by reducing the well-known Bin-Packing problem to a special case of our problem;
- We propose a greedy algorithm that can provide an approximate solution to minimize the migration cost and guarantee the security needs for a single enterprise migration problem, and prove its approximate factor;
- We also develop a genetic approach to a realistic migration problem in the context of multiple enterprises;
- We perform an extensive simulation study to evaluate the performance of the proposed solution under various settings. Our simulation results demonstrate that our algorithms outperforms the classic random assignment algorithm. In particular, the proposed solution improves the single enterprise moving cost by 53%, and the multiple enterprises cost by 66%.

The rest of this paper is structured as follows. Next section explains background and motivation, and puts our work in the context of on-going research efforts. In Section III, we formulate the security-aware migration problem, prove its NP-completeness, then propose a heuristic approach to provide an approximate solution and approximation factor, and lastly for a more general case, we provide a multi-user case. Section IV shows where the proposed algorithms play a role in the migration planning system. Section V illustrates the results of our simulation works. A discussion is presented in Section VI followed by related work VII, and we conclude the paper in Section VIII.

II. Background & Motivation

There has been a fast-growing number of enterprises that move their servers, databases, applications from existing physical/virtual machines to the cloud, because of the on-demand service and rapid elasticity offered by the cloud servers. However, there has been a debate over the cloud computing ever since the first day it was launched, where people are skeptical about using cloud due to security threats. The remote-resource-sharing feature of the cloud has posed significant challenges over the data transmission and storage. The cloud is especially vulnerable to attacks against user access and resource confidentiality as the cloud users do not have full control over network infrastructures. There has been a number of reports recently on data breaches on public/private cloud where hackers exploited the weaknesses of cloud and users' private data was leaked without detection. Security has been a deterrent to prevent enterprises from migrating into the Cloud. Among enormous security issues related to the cloud−network security, physical security, personnel security, availability, application security, privacy, and legal issues [31]− the network security is especially critical as it directly affects running services. For instance, while enterprises run a mixture of dev/ops servers and production servers, dev/ops servers tend to open ports (without security approval) for the sake of development convenience. However, a vulnerable server that opens arbitrary ports without strict security protection can expose the whole network space belonging to the same VLAN (i.e., the same subnets) because the server usually has access to other servers. Thus it is important and necessary to implement security policies in the cloud to monitor and protect services [15]. However, adding security protection introduces extra costs to users as it requires long term, constant input from cloud providers and users. To address these challenges, we aim to design a migration strategy that meets the security needs while minimizing the migration cost.

It is important to understand the design of data centers. Modern data centers such as SoftLayer mostly are organized in Pods, where each pod consists of servers, and is connected to other servers through routers/switches. A cloud data center built upon Pods can support up to 5,000 servers. This "Data Center - Pod - Server Rack" design emphasizes individuality and diversity, as servers in different Pods can be utilized for various purposes, while it still promises the capability that the cloud can work as a whole if needed. For example, SoftLayer consists of 13 data centers. Among them, the smallest data center has 5,500 servers where only one Pod is established. The biggest one has 40,000 servers that are partitioned into 8 Pods. Another advantage in this Pod design is that if multiple servers in one Pod have security needs and if they belong to the same enterprise, only one security policy needs to be deployed in the Pod, as we can implement it in the Pod switch. By leveraging this "Data Center-Pod-Server Rack" structure, we are able to optimize data center performances in terms of space, power, network, personnel, and internal infrastructure [30].

Figure 1 illustrates a simplified Pod design in data centers. Each Pod has two routers (frontend router and backend router) that divide network space into public networks (e.g., public IPs with VLAN) and private networks (e.g., 10.x.x.x IPs with VLAN), and each server has two interfaces that connect both networks. In this cloud, every Pod can manage up to 4,096 VLANs (12 bits are used for VLANs in 802.1q). The entire Pod package contains an integrated management system and an API server that manage the server racks with redundant environmental control. In addition, if a security policy such as a firewall network appliance is provisioned, the firewall will be directly connected to the frontend router and monitor all in/out

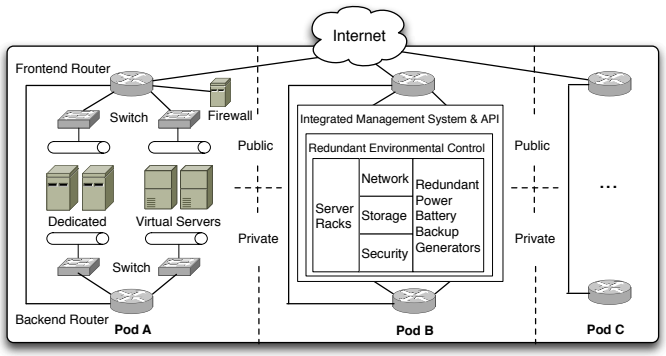


Fig. 1. Pod Design in SoftLayer Data Centers [30]

traffic within its Pod only. In addition, since the capacity of Pod is limited up to 5,000 servers, resource provisioning can fail due to the resource limit. Therefore, resource provisioning should be carefully designed in order to prevent multiple firewall network appliances from dominating the total cost.

As more and more enterprises are moving their servers, databases, applications from existing physical/virtual machines to the cloud, there have been constant migrations over data centers, leading to frequent changes in cloud settings. This has posed significant challenges in cloud management. Especially, it is incrementally hard for cloud providers to compose firewall rules every time for a new client while there exists a number of firewalls with different settings for other clients [16]. Also, in often cases, customers are left off to manually manage their own firewall appliances provisioned with no rules. Deciding security needs (e.g., how many firewall appliances are needed, where to put servers, etc.) and formulating firewall rules are hard problem to solve. Thus it is important to have a set of well-planned security rules for cloud providers to manage migration. Moreover, the migration can be costly due to the expense from the process that moves services to the cloud and the implementation of security policies. As a result, the cost reduction becomes another critical issue that enterprises need to consider when migrating.

In this paper, we address challenges arising from security implementations and cost reductions. The goal is to provide an automated planning tool that cloud providers or cloud customers can leverge to make a good migration plan considering security. In particular, we not only consider a single-customer migration scenario, but also a multiple-customer migration scenario where there exist multiple enterprises waiting to be migrated to the same data center, in various migration situations. We will formulate our problem in the next section.

III. Security-Aware Migration Algorithms

In this section we formulate the security-aware server migration problem with the objective to minimize the migration cost, and prove its NP-completeness. Then we propose heuristic approaches to provide an approximate solution and approximation factor. Lastly for a more general case, we provide a multi-user case.

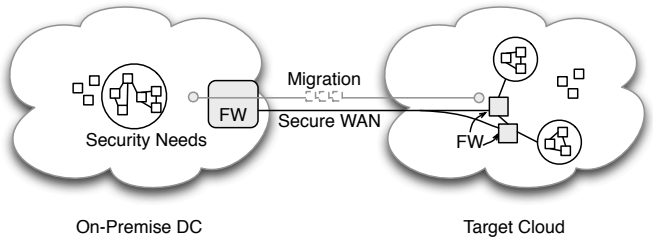


Fig. 2. Image Migration with Security Requirements (FW = firewall)

A. Problem Formulation

Figure 2 illustrates the migration process. Considering the cost and the nature of services, sensitive services (circles in Figure 2) are migrated into the cloud under the protection of extra security layers, while less sensitive components are directly integrated into the cloud and operated under cloud's default security settings. Modern clouds are organized in pods, where cloud servers in a pod are connected to the public network through a pod switch. When there exist multiple services of the same client in one server/pod with security needs, only one security policy needs to be deployed in the server/pod switch. Thereby, to reduce the migration cost due to security implementation, intuitively, we would group machines with security needs, and move them to as few servers/pods as possible in the cloud. Based on the above idea, we formulate the problem of security-aware virtual server migration in cloud computing as follows.

Consider an enterprise that runs m local machines $L_i, i \in \{1, \ldots, m\}$. Let SI_i be the size of L_i where "size" in this paper represents L_i's physical and software constraints. In particular, we focus on 3 components of SI_i that are critical to the performance of a machine: the memory size, the number of cpu cores, and the disk space. Let S_i be the memory size of L_i, CO_i be the number of cpu cores of L_i, and DS_i be the disk space of L_i. Assume there exist n servers $D_i, i \in \{1, \ldots, n\}$ organized in h pods $P_i, , i \in \{1, \ldots, h\}$ in the cloud, where each pod has n/h servers. Let CA_i be the capacity of D_i, where "capacity" refers to the physical and software capability of D_i. Similar to the local machines, in this paper, "capacity" specifically refers to the memory size, the number of cpu cores, and the disk space. Define C_i the memory size of D_i, CR_i the number of cpu cores of D_i, and DK_i the disk space of D_i. The local machine L_j can be migrated to the cloud server D_k on condition that $SI_j \leq CA_k$. Since we only consider constraints that come from memory, cpu cores, and disk space, the condition $SI_j \leq CA_k$ can be expanded as:

$$S_j \leq C_k, CO_j \leq CR_k, DS_j \leq DK_k,$$

In addition, multiple local machines $\{L_i, L_j, \ldots, L_p, i, j, \ldots, p \in \{1, \ldots, m\}\}$ can be migrated to the same cloud server $D_q, q \in \{1, \ldots, n\}$ on condition that $\sum_{k \in \{i,j,\ldots,p\}} SI_k \leq CA_q$, which can be further written as:

- $\sum_{k \in \{i,j,\ldots,p\}} S_k \leq C_q$
- $\sum_{k \in \{i,j,\ldots,p\}} CO_k \leq CR_q$
- $\sum_{k \in \{i,j,\ldots,p\}} DS_k \leq DK_q$

Let $F_i \in \{0,1\}$ represent the security need of L_i, where $F_i = 1$ indicates extra security policy needs to be implemented when migrating L_i to the cloud, and $F_i = 0$ means there are no special security needs. For instance, L_2 can be moved to D_5 when $F_2 = 1$ on condition that $SI_2 \leq CA'_5$ expanded as:

$$S_2 \leq C'_5, CO_2 \leq CR'_5, DS_2 \leq DK'_5,$$

where CA'_5 (C'_5, CR'_5, DK'_5) represents the current capacity (memory size, cpu cores, and disk space) of D_5, as the capacity of a cloud server may change constantly due to the migration of multiple servers during the entire migration process. Besides, a security policy such as firewall should be implemented in D_5 in order to meet the security need of L_2.

We further assume that if there exist multiple machines from one enterprise, all with security needs, and if they are migrated to the same server, only a single security policy needs to be implemented in the server. Moreover, if there exist multiple machines from one enterprise, all with security needs, and if they are migrated to the same pod (no need to be the same server), only a single security implementation is needed, as the cloud provider can operate the security settings in the pod switch. Let the cost of implementing a single security policy be t, where the expenses come from adding certain layers of security, software configuration and maintenance, labor work(hardware management, employee training), etc. Without loss of generality, we assume the cost of setting up a single security policy in the cloud is the same for all the clients. This can be easily extended to the case that the cost varies according to the requirements of clients.

Let A be a $m * n$ matrix that represents the migration relationship, where the element at row k and column i $A(k,i) = 1$ indicates that L_k is moved to D_i, and $A(k,i) = 0$, otherwise. Given the capacity constraint mentioned above, we can obtain:

$$\begin{aligned} \sum_{k=1}^{m} A(k,i)S_k &\leq C_i, \\ \sum_{k=1}^{m} A(k,i)CO_k &\leq CR_i, \\ \sum_{k=1}^{m} A(k,i)DS_k &\leq DK_i. \end{aligned} \tag{1}$$

In addition, assume a service can not be migrated to more than one server, we have:

$$\sum_{i=1}^{n} A(k,i) = 1. \tag{2}$$

Thus, the cost T of moving m machines to n servers is:

$$T = t \sum_{j=1}^{h} \max_{k \in \{1,\ldots,m\}, D_i \in P_j} (A(k,i)F_k) \tag{3}$$

Our goal is to move m local machines to n servers in the cloud with the minimum cost and security needs satisfied. In other words, we need to figure out a migration matrix A to solve the following problem:

$$\begin{aligned} \min_A \{ t \sum_{j=1}^{h} \max_{k \in \{1,\ldots,m\}, D_i \in P_j} (A(k,i)F_k) \} \\ \text{s.t } \sum_{k=1}^{m} A(k,i)S_k \leq C_i, \forall i, \text{ and} \\ \sum_{k=1}^{m} A(k,i)CO_k \leq CR_i, \forall i, \text{ and} \\ \sum_{k=1}^{m} A(k,i)DS_k \leq DK_i, \forall i, \text{ and} \\ \sum_{i=1}^{n} A(k,i) = 1, \forall k \end{aligned} \tag{4}$$

The above problem not only minimizes the security cost, but also leads to a minimized number of pods used for migration. This is intuitive due to the fact that when there exist multiple services (from the same enterprise) with security needs and migrated into the same pod, only a single security policy needs to be implemented (in the pod switch). Thus minimizing the security cost is equivalent to minimizing the number of pods used. We will prove in the following section that the Eq.4 is a NP-hard problem. A greedy algorithm is proposed in section III to achieve the approximate results.

B. NP Completeness

In this section we prove the security-aware migration problem is NP-hard by reducing the bin packing problem.

Theorem III.1. *Eq.4 is NP-hard.*

Proof: Consider the Bin Packing problem, where we are given m items with sizes $w_i, i \in \{1, 2, \ldots, m\}$ and a number of bins each with a capacity of c. The question is to find the minimum number of bins to place all m items. The problem is known to be NP-hard. We present in the following that, we can reduce the Bin Packing problem to a special case of the proposed migration problem Eq.4 in polynomial time.

Let's consider a special case of the proposed migration problem. Assume every server has the same capacity CA (Here capacity can refer to any of the memory, the number of cpu cores, and the disk space). Our goal is to move m local machines with sizes $\{SI_1, SI_2, \ldots, SI_m\}$ (Here size can refer to any of the memory, the number of cpu cores, and the disk space)of an enterprise to n servers with the minimized cost. Assume all m local machines have security requirements. Thus $F_k = 1, \forall k$. Consequently, Eq.4 becomes

$$\min_{A}\{t\sum_{j=1}^{h}\max_{k\in\{1,\ldots,m\},D_i\in P_j}A(k,i)\}$$
$$\text{s.t}\ \sum_{k=1}^{m}A(k,i)SI_k \le CA_i, \forall i,\ \text{and} \quad (5)$$
$$\sum_{i=1}^{n}A(k,i)=1,\forall k.$$

In particular, the term $\sum_{j=1}^{h}\max_{k\in\{1,\ldots,m\},D_i\in P_j}A(k,i)$ in Eq.5 represents the number of pods used for migration. Let $N=\sum_{j=1}^{h}\max_{k\in\{1,\ldots,m\},D_i\in P_j}A(k,i)$ be the number of pods used in the migration. Eq.5 can be further rewritten as

$$\min_{A}\{tN\}$$
$$\text{s.t}\ \sum_{k=1}^{m}A(k,i)SI_k \le CA_i, \forall i,\ \text{and} \quad (6)$$
$$\sum_{i=1}^{n}A(k,i)=1,\forall k.$$

Since t is a constant that represents the cost of implementing a security policy, in the above equation, we focus on $\min_A\{N\}$. In other words, we are looking for a migration strategy A that assigns m local machines to n servers with the minimum number of servers. As a result, finding the minimum number of bins to place all m items is equivalent to a special case of our migration problem. Through above steps, we successfully transform the Bin Packing problem to a special case of Eq.4. Therefore, Eq.4 is NP-hard. ■

C. Approximate Solution

In this section, we present a heuristic algorithm to provide an approximate solution to the proposed NP-hard problem Eq.4. In our solution, we partition machines into 2 clusters, with and without security needs based on the consideration that we attempt to implement as few security defenses as possible by grouping machines with security needs to reduce the cost. Our algorithm consists of two phases: In the first phase, we migrate machines with security needs into the cloud. In the second phase, we migrate the rest of machines to the cloud. A greedy assignment algorithm 2 based on Best-Fit is proposed for migration. We first sort machines to be migrated in decreasing order of their size SI_k (which can be expanded to memory size S_k, cpu cores CO_k, and disk space DS_k), as machines with large size should be migrated first when the resources are sufficient, otherwise, it would be hard to insert a large machine into small server when big servers are occupied /there are no big servers available. We then sort cloud servers in increasing order of their capacities CA and always start migration with the first feasible one, as we intend to fill servers/pods used before, such that we can minimize the number of servers/pod employed, which in turn will minimize the migration cost according to Eq.4. Details are described in Algorithm 1 and 2.

Algorithm 1 Cluster Migration $(\{L_i\},\{SI_i\},\{D_j\},\{CA_j\})$

Input:

- $\{L_i\}$: machines to be moved
- $\{SI_i\}$: Sizes of $\{L_i\}$, including memory size, cpu cores, and disk space
- $\{D_j\}$: servers in the cloud
- $\{CA_j\}$: Capacities of $\{D_j\}$, including memory size, cpu cores, and disk space

Output:

- A: The migration matrix
- $\{CA'_j\}$: The current available capacities of $\{D_j\}$

1: **function** CLUSTER MIGRATION ALGORITHM
2: Let A be a $m\times n$ matrix, indicating the migration relationship between $\{L_i\}and\{D_j\}$. Initially, all elements in A are zero.
3: Let $\{CA'_j\}$ represent the current capacities of $\{D_j\}$. Initially, $\{CA'_j\}=\{CA_j\}$.
4: Let Θ be the set of servers used for migration. Initially, $\Theta=\phi$.
5: Add all $L_i, i\in 1,\ldots,m$ with $F_i=1$ to Ψ, and the rest of servers to Ω.
6: Greedy Assignment$(A,\Psi,\Theta,\{CA'_j\})$
7: Greedy Assignment$(A,\Omega,\Theta,\{CA'_j\})$
8: Output A and $\{CA'_j\}$;
9: **end function**

D. Approximation Factor

In this section, we prove that the proposed cluster migration algorithm 1 can achieve an approximation factor of 2.

Lemma III.1. *There exists at most one server in Θ that has machines migrated less than half of its capacity.*

Proof: Assume there exist two servers D_i, D_j in Θ with machines less than half of their capacities. Without loss of generality, assume $CA_i \ge CA_j$. Then according to algorithm 1, items in D_j must have been migrated into D_i. Thus the assumption that there are two servers in Θ with machines less than half of their capacities does not hold, which in turn proves there exists at most one server in Θ that has machines less than half of its capacity. ■

Lemma III.2. *The sum of capacities of servers used by the optimal solution must be greater than or equal to the sum of sizes of local machines.*

Proof: Assume the sum of capacities of servers used by the optimal solution is less than the sum of sizes of local machines. Then there does not exist such a solution where we can migrate all local machines to the cloud. Therefore, the sum of capacities of servers by the optimal solution must be greater than or equal to the sum of sizes of local machines. ■

Theorem III.2. *The sum of capacities of servers used by the proposed algorithm is approximately less than twice of that used by the optimal solution.*

Proof: Based on lemma III.2, we obtain:

$$\sum_{D_i\in\Theta_{optimal}}CA_i \ge \sum_{j=1}^{m}SI_j, \quad (7)$$

Algorithm 2 Greedy Assignment Algorithm $(A, \Delta, \Lambda, \{CA'_j\})$

Input:

- A: The migration matrix
- Δ: The set of machines to be moved
- Λ: The set of servers used
- $\{CA'_j\}$: The current capacities of $\{D_j\}$

Output:

- A: The migration matrix
- Λ: The set of servers used
- $\{CA'_j\}$: The current available capacities of $\{D_j\}$

function GREEDY ASSIGNMENT
  Sort machines $\Delta = \{L_k\}$ in decreasing order of SI_k. ▷ First sort Δ by the memory size S_k. For machines with the same S_k, sort them by the number of cpu cores CO_k. Then for machines with the same CO_k, sort them by the disk space DS_k. This is because in real world applications, most of times, memory size is the bottleneck.
  for $L_k \in \Delta$ **do**
    Sort Λ in increasing order of CA'_j. ▷ Similarly, first sort Λ by the memory size C'_j. For servers with the same C'_j, sort them by the number of cpu cores CR'_j. Then for machines with the same CR'_j, sort them by the disk space DK'_j.
    Migrate L_k to the first feasible D_p in Λ. Let $A_{kp} = 1$ and update $CA'_p = CA'_p - SI_k$.
    if there is no feasible server for L_k in Λ **then**
      Sort servers that are not in Λ in decreasing order of CA'_j
      Migrate L_k to the first server D_q. Let $A_{kq} = 1$ and update $CA'_q = CA'_q - SI_k$.
      Add D_q to Λ
    end if
  end for
  Output A, Λ, $\{CA'_j\}$.
end function

where $\Theta_{optimal}$ is the set of servers used by the optimal solution. On the other hand, lemma III.1 indicates there is at most one server used by the proposed algorithm 1 that has machines less than half of its size. Let D_k be such a server. We have:

$$\sum_{j=1}^{m} SI_j > \frac{\sum_{D_i \in \Theta} CA_i - CA_k}{2} \tag{8}$$

Combine Eq.7 and 8. The following equation must hold.

$$\sum_{D_i \in \Theta_{optimal}} CA_i > \frac{\sum_{D_i \in \Theta} CA_i - CA_k}{2} \tag{9}$$

We can easily rewrite the above equation as:

$$2 \sum_{D_i \in \Theta_{optimal}} CA_i + CA_k > \sum_{D_i \in \Theta} CA_i \tag{10}$$

■

E. Multi-User Migration Algorithm

So far we have considered the problem of migrating machines of a single client to the cloud. In this section, we present that our solution can be extended easily to solve the multi-user migration problem.

Assume there are r users $\{u_1, u_2, \ldots, u_r\}$ who need to move their machines to the cloud. Let $\{L_{u_p k}, k \in \{1, \ldots, m_{u_i}\}\}$ be the set of machines of u_p. Let $\{SI_{u_p k}, k \in \{1, \ldots, m_{u_p}\}\}$ represent the sizes of machines, where m_{u_p} is the number of machines u_p has. Denote $F_{u_p k}$ the indicator of security need of $L_{u_p k}$, where $F_{u_p k} = 1$ means $L_{u_p k}$ requires protection from the security policy, and $F_{u_p k} = 1$ states $L_{u_p k}$ has no security need. Let $AR = [A_{u_1}, \ldots, A_{u_r}]^T$ represent the migration relationship between machines and servers in the cloud, where A_{u_p} is the migration matrix for user u_p. The rest of notations follows the previous sections. We are looking for a migration strategy AR that minimizes the migration cost. Following Eq.4, we can write down the objective function for the multi-user migration problem as:

$$\begin{aligned} \min_{AR} \{ t \sum_{j=1}^{h} \max_{u_p, p \in \{1,\ldots,r\}} (\max_{k \in \{1,\ldots,m_{u_p}\}, D_i \in P_j} (A_{u_p}(k,i) F_{u_p k})) \} \\ \text{s.t} \sum_{u_p} \sum_{k=1}^{m_{u_p}} A_{u_p}(k,i) SI_{u_p k} \leq CA_i, \forall i, \text{ and} \\ \sum_{i=1}^{n} A_{u_p}(k,i) = 1, \forall k \text{ and } \forall u_p \end{aligned} \tag{11}$$

In order to solve Eq.11, we design an approximate algorithm similar to the cluster migration algorithm 1. Given r users and their machines to move, we consider machines with and without security needs separately. In particular, we treat machines with security needs but from different users as different sets, while we group machines without security needs together (no matter which user they belong to). This is based on the following considerations: First, for machines from different enterprises, their security needs might be different. Therefore, they can not be processed as a whole. Second, for machines without security needs, their performances are

Algorithm 3 Multi-user Cluster Migration Algorithm $(\{L_{u_i j}\}, \{SI_{u_i j}\}, \{D_k\}, \{CA_k\})$

Input:

- $\{L_{u_i j}\}$: machines to be moved
- $\{SI_{u_i j}\}$: Sizes of $\{L_{u_i j}\}$
- $\{D_k\}$: servers in the cloud
- $\{CA_k\}$: Capacities of $\{D_k\}$

Output:

- AR: The migration matrix
- $\{CA'_k\}$: The current available capacities of $\{D_k\}$

function MULTI-USER CLUSTER MIGRATION ALGORITHM
  Let $AR = [A_{u_i}]^T$. Initially, all elements in AR are zero.
  Let $\{CA'_k\}$ represent the current capacities of $\{D_k\}$. Initially, $\{CA'_k\} = \{CA_k\}$.
  Let Θ be the set of servers used for migration. Initially, $\Theta = \phi$.
  $\forall j$, add $L_{u_i j}$ whose $F_{u_i j} = 1$ to Ψ_{u_i}, and the rest of servers to Ω.
  for all u_i **do**
    Greedy Assignment$(AR, \Psi_{u_i}, \Theta, \{CA'_j\})$
  end for
  Greedy Assignment$(AR, \Omega, \Theta, \{CA'_j\})$
  Output AR and $\{CA'_j\}$;
end function

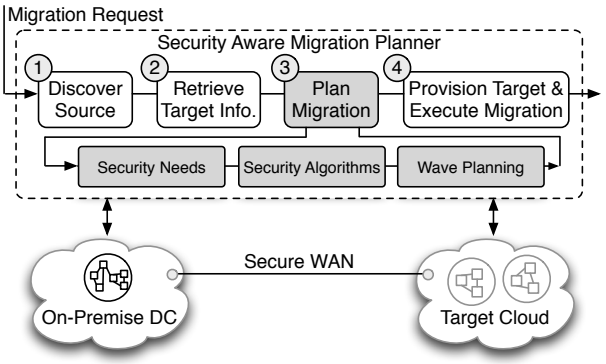
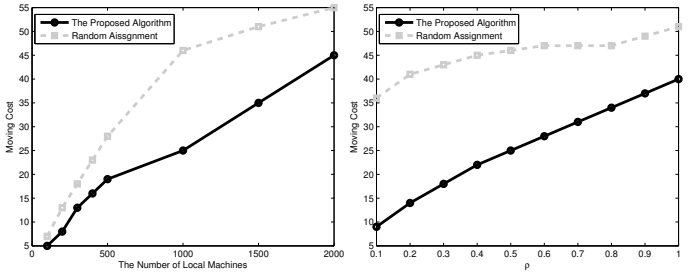


Fig. 3. Security Aware Migration Planner Architecture



(a) (b)

Fig. 4. (a) Moving Cost vs. Number of Local Machines; (b) Moving Cost vs. ρ.

affected by pods' positions in the cloud, as long as they are in the same cloud.

IV. System Architecture

In this section, we illustrate the system architecture where the proposed algorithm can be used as a part of a migration planner. Figure 3 shows a holistic migration process: 1) discover the source environment including server specifications/dependencies; 2) retrieve the target information such as available resources; 3) plan the migration considering security; 4) provision the target environment and execute the migration (assuming the secure WAN connection is established in this step). We skip the post optimization and steady-state management step in the figure, which can be achieved after finishing the entire migration process.

The goal of the migration planning step (Step 3 in Figure 3) is to automatically find the best migration planning while considering the security needs. To note, the security aware migration planner not only considers a security problem, but also takes into consideration many other problems such as target resource planning, migration wave planning, cost minimization and the like. As a migration planner built in the cloud provider's infrastructure and provided to customers, it has a global view that customers can not see. So it is more efficient that the cloud provider optimizes global resources based on the migration requests from multiple customers. With the discovered data from the on-premise datacenter, we find the servers that require security needs (normally data protection). Then, the security guaranteed optimization shown in Algorithms 1 and 3 can provide a way to create a comprehensive security aware migration planning.

V. Evaluation

In this section we validate the performance of our algorithms over simulations under various conditions.

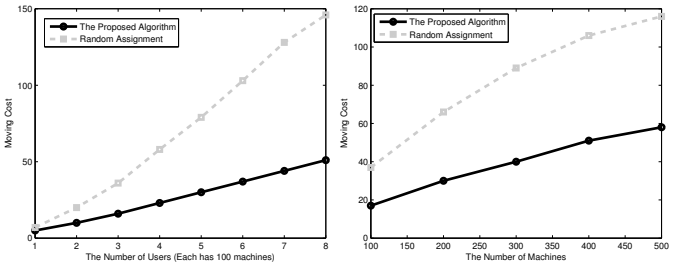
A. Evaluation Settings

Following the mainstream work [12, 35], we simulate a cloud cluster with 2000 servers organized in 50 pods, where each pod has a switch that connects to 40 servers and other pod-level switches. In our cloud, each server has 16 cpu cores, 2 TB disk space, and 128 GB memory, while, however, the actual available number of cpu cores in each server is drawn uniform-randomly from $[0, 16]$, the actual available disk space in each server is drawn uniform-randomly from $[0, 2]$ (TB), and the actual available memory is uniform-randomly from $[0, 128]$ (GB). Following notations in section III-A, let m be the number of local machines to move in an enterprise, where m varies in $\{100, 200, 300, \ldots, 2000\}$. We also draw the number of cpu cores of each local machine uniform-randomly from $[1, 16]$, the disk space uniform-randomly from $[0, 2]$ (TB), and the memory uniform-randomly from $[0, 128]$ (GB). Let ρ be the ratio of the number of local machines with security needs to m. Define $T_{total} = T + T_{extra}$, the total moving cost due to migration, where T_{extra} represents the cost of implementing security policies in extra machines when existing servers in the simulated cluster can not satisfy the user needs. Since it may occur during the migration that existing servers are not able to accept more services, as services migrated earlier occupied the resources, in order to solve this problem, we simulate a backup cloud cluster with the same size (2000 servers in 50 pods), and use it for migration when the primary cloud cluster is not able to absorb all the services. We will examine the proposed algorithm by evaluating the moving cost T_{total} under a variety of conditions.

We also implement a random assignment algorithm for performance comparison. Given 2000 servers, and m local machines, the random assignment algorithm randomly moves local machines to feasible servers. For instance, in order to move local machine l_i, the random algorithm will pick servers randomly, until it finds a server d_j such that the following conditions are satisfied: $S_i \leq C_j$, $CO_i \leq CR_j$, and $DS_i \leq DK_j$ (Similarly, if existing servers in the primary cloud can not satisfy the conditions, the backup cloud would be used).

B. Evaluation Results

1) Single Enterprise Migration: We first consider a scenario that a single client wants to move his machines to the cloud. Figure 4(a) reports the moving cost vs. the number of local machines when $\rho = 0.5$. We can observe that the cost T increases as the number of local machines to move increases. This is because when ρ is fixed, a larger number of local machines results in a larger number of machines with security needs, which in turn leads to more non-zero $A(i, j)F_i$ terms in Eq.3. Compared with the random assignment algorithm, the proposed approach achieves lower moving cost. This indicates that the proposed algorithm is cost-effective against the random assignment. Moreover, the performance gap



(a) (b)

Fig. 5. (a) Moving Cost vs. Number of Users; (b) Moving Cost vs. Number of Machines When There Are 3 Users.

between two algorithms increases with an increasing number of machines, reaches the peak when the number of machines is 1000, and slightly shrinks afterwards. This phenomenon is reasonable, because initially, as the number of machines gets bigger, the advantages of using the proposed algorithm accumulate, thus the performance gap also gets bigger. When the number of machines to move gradually approaches the capacity of the cloud (2000 servers in the cloud, and the actual available number of servers is less than 2000), both the proposed algorithm and the random algorithm tend to use all the servers in the cloud, and therefore, the difference between their performances becomes smaller. In general, the proposed algorithm outperforms the random algorithm, and achieves a 53% improvement when the number of machines is 1000.

Figure 4(b) evaluates migration algorithms when ρ varies. The number of local machines to move is fixed at 1000. As can be observed in the figure, the moving cost increases monotonically when ρ increases. This is intuitive that as ρ increases, the number of non-zero $A(i,j)F_i$ terms (Eq.3) increases, leading to a larger cost T. In general, the proposed method generates less cost under all values of ρ, which further confirm the superiority of the proposed method. Again, the performance gap shrinks as ρ increases. This is explained as follows. Given Eq.3, it is obvious that T mainly comes from machines with security needs. When ρ is small, the proposed algorithm tends to organize machines with security needs within a few number of servers, while the random algorithm simply disperses them over 2000 servers, thereby, leading to a big performance gap. When ρ climbs up towards 1, the propose method has to employ a much larger number of servers in the cloud, as the number of machines with security needs increases significantly, while the random algorithm remains basically the same, resulting in a less difference in their performance.

2) Multiple-Enterprise Migration: Next we consider a scenario that there come requests from multiple enterprises simultaneously. Following our previous notations, let r be the number of enterprises, and vary r in $\{1, 2, 3, \ldots, 8\}$. Fix ρ at 0.5, and assume each enterprise has 100 servers to move. We plot the results in Fig.5(a). First, we observe that the moving cost increases as the number of enterprises increases. This is because when r climbs up, the total number of machines with security needs increases, thus leading to a higher cost. Second, the proposed algorithm achieves lower cost than the random algorithm no matter how many enterprises exist. This again shows the superiority of our approach. Third, we observe that the performance gap between two algorithms increases significantly with an increasing number of enterprises, and

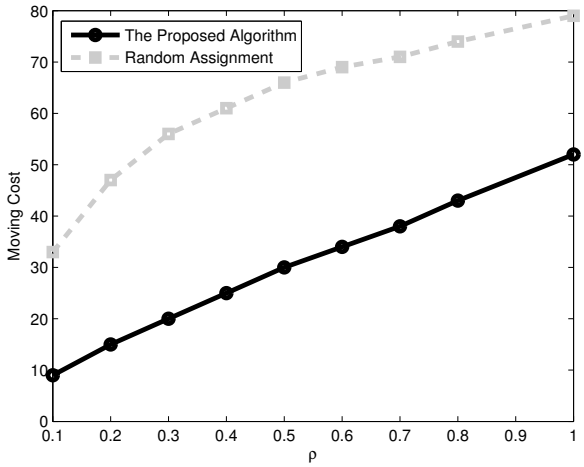


Fig. 6. Moving Cost vs. ρ.

achieves a 66% improvement when the number of enterprises reaches 8. This is due to the fact that when there exist machines whose $F_i = 1$ from multiple enterprises migrated to the same server, not one but multiple (equal to the number of enterprises that have been migrated to this server) security implementations have to be operated. Therefore, the more enterprises, the more the accumulated advantages using the proposed algorithm, the larger the gap. Our proposed method achieves an overall less migration cost in an environment close to the real world scenario where usually the cloud provider has business with hundreds of enterprises.

Fig.5(b) reports the moving cost vs. the number of local machines when there exist multiple users ($r = 3$). ρ is fixed at 0.5. In general, the proposed method outperforms the random algorithm in terms of moving cost, consistent with Fig.4(a). Also the performance gap increases as the number of machines increases. The reason is explained in Fig.5(a).

Finally, Fig.6 presents the performance of migration algorithms vs. ρ where there exist multiple enterprises ($r = 3$). Each enterprise has 200 machines to move. Results demonstrate the superiority of our algorithm when there exist multiple customers waiting to be migrated. This is consistent with Fig.4(b). The above results show that the proposed method has a strong ability to reduce the moving cost and outperforms the random algorithm under various conditions.

VI. Discussion

An enterprise-scale migration consists of multiple steps including source discovery, pre-migration wave planning, migration, post-migration configuration/optimization, and steady-state operations. However, existing tools are not able to manage these migration steps as a whole, as it is really hard to accommodate various requirements from each step. For example, the image-based migration software, Racemi, while it supports P2V, V2V, and P2P services[1] on both Windows and Linux operating systems, has not been working on other popular platforms, such as Windows 2000 and 2012. Thus, in this

[1]P stands for physical and V stands for virtual. Converting means changing image formats or adapting a server to the target (e.g., removing drivers that access hardwares directly).

section, we discuss about issues and limitations rising from cloud migrations, and future directions.

Cloud-born vs. migration: The successful story, among many, of Netflix open source software (OSS)−100s of mid-tier services and applications, billions of requests per day, up to 70 billion events per day, 10,000s of EC2 instances in multiple regions−ignites flame on cloud-based business models. Yet the Netflix OSS is a cloud-born architecture considering resource elasticity and location-based replications, whereas the existing on-premise data centers often do not have a cloud-fit architecture. Enterprises have been seeking ways to adapt their environment to the cloud and migrate into the cloud instead of re-architecting the IT infrastructure in the cloud. A fundamental difference between on-premise data centers and the cloud environment is that the cloud follows the pay-as-you-go model, which means customers need to manage computing resources efficiently. Otherwise a monthly payment may include unnecessary bills. This is essentially problematic if we migrate the on-premise data centers as they are because they are designed based on peak performance usage. Also an entire on-premise data center is protected by a firewall, but in the cloud environment, the servers may have to be distributed across regionally separated data centers or Pods in a data center that change the network security configurations.

Multiple cloud providers: Since Amazon launched the AWS service in 2006, many similar Infrastructure-as-a-Service (IaaS) companies have jumped in the same business model. This means that customers can choose cloud providers based on their requirements or preferences. Accordingly, there needs to be a cloud moving company that can pack all the resources from one cloud and unpack in another cloud to make them operate properly. Unfortunately the current migration offerings are not sufficient to satisfy this demand because they move one virtual server basis and the reconfiguration is left to the customers.

Migration orchestrator: As explained before, an entire migration process is a long lasting task that involves many steps−source discovery, discovery data analytics and scheduling, target provisioning, source-target secure network setup, migration, and remediation and test [14]−and it may take weeks, even months based on a number of servers and security requirements. Currently many different companies offer piece(s) of the entire migration process, so that it is really hard to manage the end-to-end migration process. There is certainly a demand to orchestrate the end-to-end migration process from discovering source environment and optimize the target resources after the migration.

To sum up, an end-to-end migration needs a holistic planning, spanning from security to resource management with a goal to achieve the architecture that can leverage an elastic resource scalability and regional deployment across data centers in one cloud provider or multiple cloud providers. While currently there are many separate migration researches, there needs to be a migration engine that should combine them together to provide an end-to-end migration orchestration. While the end-to-end migration process often requires not only automated processes and human-driven activities, using a business process management software may help to streamline the entire migration process [24, 25]. As a part of the migration problem solution, our contribution in this paper can be applied to the migration planning stage in that the objective is to minimize the extra resources and implementation cost.

VII. Related Work

This paper is built upon considerable prior work regarding data center migration, and client-server assignment in cloud computing. In this section, we review the related work in each of these fields.

There has been a growing body of research on data center migration, where energy minimization [34, 4, 6, 4] and bandwidth utilization maximization [26, 33, 36, 17] were key areas that have been focused on. Recent work on energy minimization mainly reduce energy consumption by utilizing the market-based resource allocation, where the theory of supply and demand in economics was introduced to help achieve the goal. In [8, 7], authors propose such economical models based on data center usage expectations and energy costs, which requires accurate usage prediction, and efficient information exchange between enterprises and cloud-providers. On the other hand, a number of works discuss the problem of migration with bandwidth guarantees. Migration strategies in these work were designed with awareness of the network topology connecting servers in the cloud, in order to avoid load imbalance. For example, [26] presented a solution to the bandwidth-aware migration problem based on the maximum-flow tree, where the tree nodes represent the servers, and edges stand for traffic. While these work achieved improvements in either power consumption or network utilization, few of them have taken the security need of data centers into account. This paper addresses the challenge of security-aware data center migration, where we aim to minimize the migration cost without compromising the security of data centers. For the first time, we introduce the security cost into the data center migration, in responding to the urgent request for security implementations from both individuals and organizations.

The client-server assignment problem in cloud computing has led to extensive discussions recently. Current work on client-server assignment can be classified into three categories: 1. assignments with an emphasis on load minimization [3, 11, 32], 2. assignments with an emphasis on load balancing [2], and 3. a hybrid of the former two [23, 37, 5]. Prior work that falls into the first category, investigated the assignment problem by representing clients and servers as nodes, and communications as edges in a graph. A clustering algorithm was applied partitioning the graph such that the total communication would be minimized. Examples of the second category includes [20, 21], where a similar idea based on clustering algorithms was adopted to solve the balanced-clustering problem over graphs. Authors in [23] considered both communication load and load balance among servers. The paper modeled the client-server assignment as a convex optimization problem, and employed a heuristic algorithm for approximate solution. Our approach in this paper is similar to the client-server assignment in that while the total communication load is minimized in the assignment problem, we consider the moving cost is minimized. However, our work is different from the assignment problem because we focus on minimizing

the cost of individual enterprise, rather than the total cost of the entire cloud.

VIII. Conclusions

In this paper, we investigated the problem of virtual server migration in cloud computing where enterprises move their servers, databases, applications from existing physical/virtual machines to the cloud. In response to the growing concern of network security among cloud users, we focused on satisfying the security needs of customers while minimizing the moving cost introduced by the implementation of security policies in the cloud. We proved that the problem considered in this paper is NP-hard, and proposed a greedy algorithm with an approximate factor of 2. The proposed approach was evaluated over extensive simulations. The results demonstrated that our method outperforms the random migration algorithm in terms of cost without compromising the security needs of customers. Our future work includes detailed studies on computing the influence from the cloud topology, server load on server migration and deriving variations of the proposed strategy, i.e., for run-time application migration in mobile cloud.

References

[1] S. Al-Haj, E. Al-Shaer, and H.V. Ramasamy. Security-aware resource allocation in clouds. In *Services Computing (SCC), 2013 IEEE International Conference on*, pages 400–407, June 2013.

[2] Baruch Awerbuch, Mohammad T Hajiaghayi, Robert D Kleinberg, and Tom Leighton. Online client-server load balancing without global information. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 197–206. Society for Industrial and Applied Mathematics, 2005.

[3] Thomas Bauer and Peter Dadam. Efficient distributed workflow management based on variable server assignments. In *Advanced Information Systems Engineering*, pages 94–109. Springer, 2000.

[4] Anton Beloglazov and Rajkumar Buyya. Energy efficient resource management in virtualized cloud data centers. In *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 826–831. IEEE Computer Society, 2010.

[5] Rajesh D Bharati, Ishan Naidu, Anurag Kiran, Kalpesh Khune, and Chirag Vyas. An enhanced client-server assignment for internet distributed systems. *International Journal of Engineering Trends and Technology (IJETT)*, 2014.

[6] Niv Buchbinder, Navendu Jain, and Ishai Menache. Online job-migration for reducing the electricity bill in the cloud. In *NETWORKING 2011*, pages 172–185. Springer, 2011.

[7] Rajkumar Buyya, Anton Beloglazov, and Jemal Abawajy. Energy-efficient management of data center resources for cloud computing: A vision, architectural elements, and open challenges. *arXiv preprint arXiv:1006.0308*, 2010.

[8] Rajkumar Buyya and Manzur Murshed. Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing. *Concurrency and computation: practice and experience*, 14(13-15):1175–1220, 2002.

[9] 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)*, pages 45–56, Philadelphia, PA, June 2014. USENIX Association.

[10] DoubleTake. http://www.visionsolutions.com/products/dt-move.aspx. 2013.

[11] Umar Farooq and John Glauert. Ara: An aggregate region assignment algorithm for resource minimization and load distribution in virtual worlds. In *Networked Digital Technologies, 2009. NDT'09. First International Conference on*, pages 404–410. IEEE, 2009.

[12] Chuanxiong Guo, Guohan Lu, Helen J Wang, Shuang Yang, Chao Kong, Peng Sun, Wenfei Wu, and Yongguang Zhang. Secondnet: a data center network virtualization architecture with bandwidth guarantees. In *Proceedings of the 6th International COnference*, page 15. ACM, 2010.

[13] Mohammad Hajjat, Xin Sun, Yu-Wei Eric Sung, David Maltz, Sanjay Rao, Kunwadee Sripanidkulchai, and Mohit Tawarmalani. Cloudward bound: Planning for beneficial migration of enterprise applications to the cloud. *SIGCOMM Comput. Commun. Rev.*, 40(4):243–254, August 2010.

[14] Jinho Hwang, Yun-Wu Huang, Maja Vukovic, and Nikos Anerousis. Enterprise-scale cloud migration orchestrator. *IFIP/IEEE International Symposium on Integrated Network Management*, 2015.

[15] Ponemon Institute. Data breach: The cloud multiplier effect. 2014.

[16] Sotiris Ioannidis, Angelos D. Keromytis, Steve M. Bellovin, and Jonathan M. Smith. Implementing a distributed firewall. In *Proceedings of the 7th ACM Conference on Computer and Communications Security*, CCS '00, pages 190–199, New York, NY, USA, 2000. ACM.

[17] Navendu Jain, Ishai Menache, Joseph Seffi Naor, and F Bruce Shepherd. Topology-aware vm migration in bandwidth oversubscribed datacenter networks. In *Automata, Languages, and Programming*, pages 586–597. Springer, 2012.

[18] Jill Jermyn, Jinho Hwang, Kun Bai, Maja Vukovic, Nikos Anerousis, and Salvatore Stolfo. Improving readiness for enterprise migration to the cloud. In *Proceedings of the Middleware Industry Track*, Industry papers, pages 5:1–5:7, New York, NY, USA, 2014. ACM.

[19] Joanna Kolodziej, Samee Ullah Khan, Lizhe Wang, Marek Kisiel-Dorohinicki, Sajjad A. Madani, Ewa Niewiadomska-Szynkiewicz, Albert Y. Zomaya, and Cheng-Zhong Xu. Security, energy, and performance-aware resource allocation mechanisms for computational grids. *Future Gener. Comput. Syst.*, 31:77–92, February 2014.

[20] Kevin Lang. Finding good nearly balanced cuts in power law graphs. *Manuscript*, 2004.

[21] Kevin Lang. Fixing two weaknesses of the spectral method. In *Advances in Neural Information Processing Systems*, pages 715–722, 2005.

[22] Michael Nidd, Kun Bai, Jinho Hwang, Maja Vukovic, and Michael Tacci. Automated business application discovery. *IFIP/IEEE International Symposium on Integrated Network Management*, 2015.

[23] Toyoaki Nishida. *An Engineering Approach to Conversational Informatics*. Springer, 2012.

[24] IBM International Technical Support Organization. Ibm business process manager version 8.0 production topologies. *IBM Redbooks*, 2013.

[25] IBM International Technical Support Organization. Websphere application server v8.5 concepts, planning, and design guide. *IBM Redbooks*, 2013.

[26] Jing Tai Piao and Jun Yan. A network-aware virtual machine placement and migration approach in cloud computing. In *Grid and Cooperative Computing (GCC), 2010 9th International Conference on*, pages 87–92. IEEE, 2010.

[27] Racemi. http://www.racemi.com/. 2013.

[28] Rackware. http://www.rackwareinc.com/. 2014.

[29] Jia Rao, Xiangping Bu, Cheng-Zhong Xu, Le Yi Wang, and Gang George Yin. VCONF: a reinforcement learning approach to virtual machines auto-configuration. In *Proceedings of the 6th International Conference on Autonomic Computing, ICAC 2009, June 15-19, 2009, Barcelona, Spain*, pages 137–146, 2009.

[30] SoftLayer. http://www.softlayer.com/data-centers. 2014.

[31] Wikipedia. Cloud computing security. In *http://en.wikipedia.org/wiki/Cloud_computing_security #Dimensions_of_cloud_security*, 2014.

[32] Lan Xiao, Qiwen Huang, Veronica Yank, and Jun Ma. An easily accessible web-based minimization random allocation system for clinical trials. *Journal of medical Internet research*, 15(7), 2013.

[33] Yanjue Xu and Yuji Sekiya. Scheme of resource optimization using vm migration for federated cloud. *Proceedings of the Asia-Pacific Advanced Network*, 32:36–44, 2011.

[34] Kejiang Ye, Dawei Huang, Xiaohong Jiang, Huajun Chen, and Shuang Wu. Virtual machine based energy-efficient data center architecture for cloud computing: a performance perspective. In *Proceedings of the 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing*, pages 171–178. IEEE Computer Society, 2010.

[35] Mohamed Faten Zhani, Qi Zhang, Gwendal Simona, and Raouf Boutaba. Vdc planner: Dynamic migration-aware virtual data center embedding for clouds. In *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, pages 18–25. IEEE, 2013.

[36] Jie Zheng, Tze Sing Eugene Ng, and Kunwadee Sripanidkulchai. Workload-aware live storage migration for clouds. 2011.

[37] Yuqing Zhu, Weili Wu, J. Willson, Ling Ding, Lidong Wu, Deying Li, and Wonjun Lee. An approximation algorithm for client assignment in client/server systems. In *INFOCOM, 2014 Proceedings IEEE*, pages 2777–2785, April 2014.