Abstract—For economic benefits and efficient management of resources, organizations are increasingly moving towards the paradigm of “cloud computing” by which they are allowed on-demand delivery of hardware, software and data as services. However, there are many security challenges which are particularly exacerbated by the multitenancy and virtualization features of cloud computing that allow sharing of resources among potentially untrusted tenants in access controlled cloud datacenters. This can result in increased risk of data leakage. To address this risk vulnerability, we propose an efficient risk-aware virtual resource assignment mechanism for clouds multitenant environment. In particular, we introduce the notion of sensitivity in datacenters and the objective is to minimize the risk of data leakage. In addition, the risk should not exceed in high sensitivity datacenters in comparison to low sensitivity datacenters. We present three assignment heuristics and compare their relative performance.

Index Terms—Cloud services, access control, risk assessment, vulnerability

1 INTRODUCTION

Clouds allow on-demand delivery of software, hardware, and data as services. Services are made available to users on demand from a cloud providers servers and hence organizations save the cost of building and maintaining their own on-premise servers and pay only for the services they actually use. Cloud services can be dynamically scaled to meet the needs of its users and are designed to provide easy and scalable access to applications. However, the security challenges in cloud computing present a major obstacle for adopting the cloud computing paradigm [1]. The number of cloud security related incidents are on the rise [2]. Cloud Security Alliance (CSA)\(^1\) identified the top security threats in cloud computing for 2013 as data breaches resulting from vulnerability of shared virtual resources. An example of data breach is the vulnerability of Google Docs where many documents are exposed to unauthorized users [3]. The cloud providers usually implement resource isolation mechanisms to counter the risk of data leakage and to increase the resource utilization at the same time. However, resource sharing remains the main security concern of cloud customers [4]. Therefore, security-aware resource scheduling is needed in order to minimize the risk to which cloud customers application services are exposed in a multitenant cloud environment. In this paper, we propose an efficient risk-aware virtual resource assignment mechanism for multitenant cloud environment.

\(^1\) https://cloudsecurityalliance.org/research/top-threats/

1.1 Multi-tenant Cloud System

We take the perspective of Software as a Service (SaaS) provider who has its application services hosted by the Infrastructure as a Service (IaaS) providers. A SaaS provider may choose different IaaS providers to increase its service reliability by avoiding single IaaS failure. In Figure 1a, we provide a system view of the cloud architecture. The consists of a Virtual Resource Manager (VRM), a datacenter, and an Access Control Module (ACM). We can have PaaS providers instead of IaaS providers but this option can restrict the SaaS provider to host only those applications that can be deployed in PaaS. In contrast, IaaS is more flexible as it allows the selection of different types of VM images [1].

Since cloud is a multi-tenant environment, it requires security isolation among its tenants. Generally, four types of security isolation mechanisms are used for cloud architecture as depicted in Figure 1b. The first type of isolation is physical isolation where services/applications are hosted on separate physical resources. The SaaS provider achieves this isolation either by using different IaaS providers or different private clouds for a given IaaS provider [5]. The second type of isolation is at the hypervisor level where by a SaaS provider guarantees separate virtual machines (VMs) for every tenant. However, VMs may be collocated in the same physical server. The third type of isolation uses a container-based approach at the OS level where tenant applications run as separate containers sharing a common OS [6]. The fourth type of isolation is the shared service isolation where two or more tenants require a common application which is isolated among the tenants. The shared service isolation type is enforced by the application running on top of one or more VMs. In this paper, we don’t consider container-based isolation because we assume that a VM can host only one service at a time as mentioned in [7]. The cloud architecture in Figure 1a depicts the other three types of isolations. In this
paper, we do not address the problem of VM isolation. But rather, given the VM isolation techniques implemented by IaaS providers which can vary in their isolation guarantees [5], our objective is to manage the virtual resource, such that, if a vulnerability weakens the isolation among VMs, then the total risk of data leakage is minimized. If the VM isolation becomes stronger, the risk decreases but it does not vanish.

As noted in Figure 1a, the SaaS provider uses a service directory to control the service subscription where a user accesses a service only if he has the subscription to that service [8]. In addition, the SaaS provider controls its data operations using a Role-Based Access Control (RBAC) policy. Under this policy, a user of a service can perform an action in the datastore if the user is assigned a role and the role has the authorization to perform that action. As shown Figure 2a, the RBAC policy is enforced by the SaaS provider to protect the backend datastore. We assume that the roles of the RBAC policy can share data thus constituting a multi-tenant datastore environment. A role can access data within an enterprise or across enterprises under some Service Level Agreement (SLA) [9]. Collectively, the intra- and inter-enterprise data accesses by various roles can be governed by a federated RBAC policy. The SaaS provider implements the authorization mechanisms based on this federated policy. A leading practical example of using RBAC in cloud is Microsoft Azure, which uses RBAC to authorize access to its resources through the Azure Active Directory. In addition, Azure controls its data operations using RBAC for performing create/read/update/delete operations in a SQL DB [8]. Also, Oracle Sales Cloud has proposed using RBAC to secure access to its functionalities and data [10]. We only consider RBAC policy to authorize the accesses to the datastore and not the services as shown in Figure 2a.

For a datacenter residing with a SaaS provider, it is imperative that users are allowed access to multiple services based on service subscription. Such services are deployed on compatible VMs provided by IaaS providers. Due to the heterogeneity of SaaS services such as computational intensive services or data intensive services, SaaS requires different types of VMs. For such environment, the compatibility between the services and the VMs is needed. In summary, these requirements can be stated as “A role requires multiple services and each service requires compatible VMs” as depicted in Figure 2a. However, sharing of data among potentially untrusted multiple tenants, or multiple roles in our case, can cause the risk of data leakage due to cloud vulnerabilities. We define a bi-tuple/pair of a service and its compatible VM as Integrated Service-aware Virtual Machine (ISVM). We assume data can be leaked across VMs collocated in the same physical server. Such type of data leakage is due to inter-ISVMs vulnerabilities. In addition, the data leakage can happen in the SaaS provider applications shared by different tenants and such data leakage is due to intra-ISVM vulnerabilities. In Section 6, we consider two vulnerability models to quantify the probability of data leakage associated with both intra- and inter-ISVM vulnerabilities.

In [9], we have proposed a distributed access control architecture for which a key component is the VRM. The VRM allocates the virtual resource to services to satisfy some SLA requirements for each cloud user and to minimize the cost of provisioning for the SaaS providers. As illustrated in Figure 1a, VRM consists of workload estimation, resource vulnerability estimation, service compatibility and vulnerability estimation, and resource assignment components. The workload estimation component estimates the sharing ability estimation, and resource assignment components. The virtual resource vulnerability estimation component uses security analysis tools [11] to estimate the virtual resource vulnerability. Subsequently, this component can be used to characterize a virtual resource with respect to different vulnerability security measurements e.g. a highly secure or an insecure virtual resource. The service compatibility and vulnerability estimation component maintains a vendor supplied service-VM compatibility matrix and estimates the ISVM vulnerabilities as discussed in Subsection 1.2. The resource assignment component uses the workload and vulnerability estimations to assign virtual resources to cloud users application services with a goal to minimize the total risk of data leakage.

1.2 Cloud Vulnerability and Risk

The security vulnerabilities of a cloud system are considered cloud specific if they result from core cloud computing technologies. Among others, VM escape and SQL injection
vulnerabilities in virtualization and web application technologies, respectively, are considered cloud vulnerabilities [12]. The VM escape vulnerability is a weakness of security isolation (Type 2) in the virtualization layer that might allow an attacker to escape from a local VM to a collocated tenant’s VM. Since IaaS offering is based on the virtualization technology, the SaaS or PaaS providers, built on top of IaaS, are exposed to the same type of vulnerabilities. The SQL injection vulnerability is a weakness of security isolation (Type 4) in SaaS applications that can allow attackers to have unauthorized access to datastore. Physical isolation among tenants can reduce the data leakage risk. However, this isolation mechanism increases the SaaS deployment cost resulting in underutilization of its resources [13].

According to NIST [14], the security risk can be described as the loss of integrity, loss of availability, and loss of confidentiality. The loss of confidentiality can result in loss of public confidence, embarrassment, or legal action against the organization. In this paper, we focus on the risk of loss of confidentiality which can range from jeopardizing the national security to disclosure of the Privacy Act data. To quantify this risk which is due to data leakage in SaaS datacenter as: “the potential that a given threat will exploit vulnerability of an asset or group of assets and thereby cause harm to the organization”. The quantitative risk is the product of the likelihood of the attack and its consequences or impact [15]. The likelihood of an attack depends on the vulnerabilities of the system and the attacker threat. The consequence of data leakage attack is measured in terms of the size of tenant’s data asset which is leaked. Accordingly, we can formulate the risk due to data leakage in SaaS datacenter as:

\[
\text{Risk} = \text{Assets} \times \text{Vulnerability} \times \text{Threat}
\]

where \(\text{Threat} = 1\)  

We consider the cloud tenant’s assets to be its data stored in the cloud datacenter. The assets are similar for both cloud computing and conventional systems. However, the main difference between the risk in cloud and in conventional systems is due to cloud specific security vulnerabilities as mentioned above. The vulnerability in Equation 1 is the overall probability of data leakage as a result of cloud system software (i.e. application and virtualization) vulnerabilities. There are many vulnerabilities in the cloud system software that can cause data leakage across tenants e.g. VM escape, SQL injection, and cross-site scripting. To find the overall probability of data leakage across tenants, we first identify the software vulnerabilities that can cause data leakage. Such vulnerabilities have different levels of criticality which are calculated based on Common Vulnerability Scoring System (CVSS) scores [16]. Consequently, the probabilities of finding each of these vulnerabilities are estimated using statistical methods such as Vulnerability Discovery Models (VDMs) [17]. Each vulnerability probability can be weighted based on its criticality and used to find the overall probability of data leakage (vulnerability in Equation 1). Various aggregation approaches can be used to capture the overall probability of data leakage such as taking the maximum (Weakest Link Model) and the semantic-based aggregation (Dependence Model) as discussed in Section 6. The level of threat of an attacker in the risk equation depends on his/her experience, motivation, and time of access to the asset. There is no quantifiable measurement for the attacker experience or motivation and therefore, we assume all attackers pose the same threat.

**Motivating Scenario:** In a manufacturing company, various departments such as finance, engineering, human resources, and marketing as well as external contractors outsource their data to a SaaS provider and collaborate on a certain project. This project may require sharing of multiple sensitive documents among these various departments and the contractors as well as among the contractors themselves. The data sharing is controlled by a federated RBAC policy [9]. Users from various departments and from external contractors are assigned roles according to the RBAC policy. These roles access project data and process the data through project software packages provided by the SaaS provider as shown in Figure 2b. Contractors can be competitors and hence can be viewed as untrusted tenants. We assume the data from the parent company and its contractors is static and preloaded in a cloud datacenter. This scenario constitutes an RBAC multi-tenant cloud environment. In this environment, intra- and inter-IVSM vulnerabilities pose a risk of data leakage within the parent company and among its contracts.

In this paper, we propose the notion of the sensitivity of multi-tenant datacenters in terms of degree of data sharing among tenants. For example, limited sharing implies a high sensitivity datacenter and high sharing of data means low sensitivity datacenter. The goal of this paper is to present...
virtual resource management methodologies such that Risk (Equation 1) is minimized. In addition, it is desirable that for a given size of a datacenter, Risk should not exceed if the datacenter has high sensitivity as compared to the case if the datacenter has low sensitivity.

The paper is organized as follows. Section 2 discusses related work. Section 3 provides an overview of the RBAC model and defines RBAC policy and spectral model. Section 4 proposes a workload approximation model based on a given RBAC policy. Section 5 characterizes the cloud datacenter by introducing the notion of sensitivity. Section 6 discusses the key models from literature to quantify ISVM vulnerabilities. Section 7 presents the risk-aware scheduling problem and the cost functions. Section 8 presents assignment heuristics for ISVM allocation. Section 9 compares the performance of the heuristics. Finally, Section 10 outlines the conclusion. Proofs of all theorems and lemmas are given in the Appendix.

2 RELATED WORK

Security-aware Scheduling: Security-aware scheduling has been mostly reported in the literature in the context of task scheduling in untrusted computer grid systems. In [18] the authors have proposed an availability-aware secure scheduling algorithm for multi-class applications with varying availability and performance requirements. The proposed algorithm provides high availability with the cost of increasing the response time. In [19], the authors have proposed a secure scheduling algorithm based on the trustworthiness measurements of the computing node in computer grid system. The application security overhead and risk probability are computed and incorporated in the scheduling decision. In addition, a risk-resilient scheduling algorithm is proposed in [20] which ensures secure grid job execution by requiring grid node security index to be higher than user job trust requirement. In [21], the authors have presented a security-aware resource allocation mechanism for real-time applications in the case of homogeneous and heterogeneous platform. The above works do not consider the data confidentiality of application or the access control policy, which restricts permissions for a given task. A key feature that differentiates our scheduling algorithms from other approaches is that the privilege set for each task is the main parameter considered in the scheduling decision.

Access Control in Cloud Datacenter: Several researchers have addressed access control issues for cloud computing. Nurmi et al. [22] have provided an authorization system to control the execution of VMs to ensure that only administrators and owners could access them. Berger et al. [23] have proposed an authorization model based on both RBAC and security labels to control access to shared data, VMs, and network resources. Calero et al. [24] have presented a centralized authorization system that provides a federated path-based access control mechanism. What distinguishes our work is that we address the problems of virtual resource vulnerabilities in the presence of multi-tenancy and virtualization.

Vulnerability Models: VDMs are used to estimate the probability of software vulnerability. The authors in [25] have proposed an attack surface metric that measures the security of software by analyzing its source code and find the potential flows. In [17] the vulnerability discovery is modeled based on the empirical analysis of history of actual vulnerability dataset and followed by predicting the vulnerability discovery. The source code quality and complexity are incorporated in the VDM to generate better estimation [26]. CVSS [27] metrics have been extensively used by both industry and academia to score the vulnerabilities of the systems and estimate risks. The US National Vulnerability Database (NVD) provides a catalog of known vulnerabilities with their CVSS scores.

3 ROLE BASED ACCESS CONTROL MODEL

The RBAC policy defines permissions on objects based on roles in an organization. The policy is composed of a set of Users (U), a set of Roles (R), and a set of Permissions (P). UA is a user-to-role assignment relation, and PA is a role-to-permission assignment relation. An RBAC access control policy for a cloud datacenter defines permissions to access data objects for its roles [28]. Such assignment is formally defined as follow.

**Definition 3.1.** Given an RBAC policy \( P \) for a datacenter where \( R \) is the set of roles and given \( O \) being the set of data objects, the role-to-permission assignment \( PA \) can be represented as a directed bipartite graph \( G(V,E) \), where \( V = R \cup O \ s.t. \ R \cap O = \phi \). The edges \( e_{r,o_j} \in E \) in \( G \) represents the existence of role-to-permission assignment \((r_i \times o_j) \in PA \) in the RBAC policy \( P \), where \( r_i \in R \) and \( o_j \in O \).

The out-degree of a role vertex represents the cardinality of the role and the in-degree of a data object vertex represents the degree of sharing of that object among roles. In Figure 3a, an RBAC policy with \( |R| = 4 \) and \( |O| = 10 \) is represented as bipartite graph model. We can notice that the cardinality of role \( r_1 \) is \( \text{out-degree}(r_1) = 6 \). Also, the degree of sharing of data object \( o_{10} \) is \( \text{in-degree}(o_{10}) = 4 \).

The resource assignment component of VRM requires the cardinality of shared data objects among roles, as discussed in Section 7. For datacenters, determining these cardinalities from the bipartite graph is a computational intensive task. We propose an alternative representation of RBAC by clustering all data objects that are accessed by the different roles into a set of non-overlapping partitions. The set of cardinalities of these partitions, called \( W \), is termed as **spectral model** for RBAC policy. This model is formally defined as follows.

**Definition 3.2.** (RBAC spectral model). Given a bipartite graph representation of RBAC policy \( G(V,E) \), let \( P(R) \) be the power set of \( R \) excluding the null set \( \phi \). The spectral representation of RBAC is the vector \( W \), indexed by \( P(R) \) and lexicographically ordered. Formally, let \( p \in P(R) \) be a set of roles, then \( w_p \in W \) is defined as follows:

\[
w_p = |\{o_k : o_k \in O \land \forall r_i \in p \exists e_{r_i,o_k} \in E\}|
\]

and \( |W| = 2^n - 1 \).

The spectral model provides two advantages over the bipartite graph model. First, it can be used to characterize an RBAC policy in terms of **sensitivity** of a datacenter using...
a single parameter. Based on the degree of sharing among roles, the datacenter sensitivity can be high, medium, or low as elaborated later in Section 5. In addition, the spectral model allows resource assignment based on a given percentage of datacenter. Varying this percentage can lead to variable complexity of an assignment algorithm.

The set $\mathcal{W}$ can be generated from the bipartite graph model of RBAC. The members $w_p \in \mathcal{W}$ are non overlapping and can be viewed as vertices of a lattice (i.e. binary $n$-cube) with $n$ levels, where $n$ is the number of roles. For example nodes at level 1 of the lattice represent the cardinalities of partitions corresponding to unshared data objects belonging to individual roles. The nodes at level 2 represent the cardinalities of data partitions that are accessed by two roles. Similarly, the nodes at level $n$ contain data objects shared by all the roles. The nodes of $\mathcal{W}$ can be indexed using the role IDs associated with the partition as shown in Figure 3b. It can be observed that indices of $\mathcal{W}$ vector are subsets of $P(R)$ and its elements are the cardinalities of partitions that can be accessed by all the roles in these subsets. Note, $\sum_{w_p \in \mathcal{W}} w_p$ equals to the total size of datacenter. The following example illustrates the spectral model.

**Example 3.1.** In Figure 3a, an access control policy with $|R| = 4$ and $|O| = 10$ is shown as a bipartite graph.

The spectral model is shown as lattice in Figure 3b. Notice that $w_{(1,4)} = \{|o_5, o_6|\} = 2$ because $o_5$ and $o_6$ are accessed by both $r_1$ and $r_4$.

We now define the Attackability (AT) of a spectral partition. Attackability measures the maximum amount data that the adversaries (roles) can potentially access in an unauthorized manner. Formally, this parameter is defined as follows.

**Definition 3.3.** Given spectral representation of access control $\mathcal{W}$, the Attackability of $w_p \in \mathcal{W}$ is:

$$AT(w_p) = w_p \times (n - |p|)$$

**Example 3.2.** In Figure 3b, the Attackability for $w_{(2)}$ and $w_{(1,2,4)}$ is as follows:

$$AT(w_{(2)}) = 2 \times 3 = 6; \quad AT(w_{(1,2,4)}) = 1 \times 1 = 1$$

We can notice that as more roles have access permissions of a partition, its Attackability decreases. Note, that in the spectral representation of RBAC, the tuples with highest AT belong to a partition in level $\frac{n}{2}$.

### 4 Workload Estimation for RBAC Policy

For a datacenter, specifying the exact value of $w_p$'s can be computationally intensive. One practical approach is to use cardinality estimation techniques [29]. For example, for a transactional workload, the selectivity estimation of query processing for a datacenter can be used for query size estimation [29], whereby query (or a collection of queries) can correspond to a role. In case role mining is used to design a RBAC policy, the role mining technique such as multi-assignment clustering [30] can provide an estimate of $\mathcal{W}$. In this paper, we assume the access of data objects in a datacenter follows a Zipfian distribution, an assumption that is supported by Yahoo! Cloud Serving Benchmark (YCSB) [31]. YCSB uses five types of workload to benchmark different cloud data serving systems. For four of these workloads, it is assumed that data access follows the Zipfian distribution. Only one of the workloads uses the Lates distribution which is similar to the Zipfian distribution except that the most recently inserted records are in the head of the distribution.

Motivated by this real world benchmark, our experiment workflow follows Zipfian distribution. According to this distribution, some objects are shared by a large number of roles (queries) while most of the objects are shared among a fewer number of roles (queries). Hence, this distribution can provide a heterogeneous workload for RBAC. The Zipfian distribution is given as follows:

$$f(\alpha; s, N) = \frac{\alpha^{-s}}{\sum_{i=1}^{N} i^{-s}}$$

**Equation 3**

Where, $N$ : maximum rank, $\alpha$ :selected rank, $s$ : sensitivity parameter.

In Equation 3, if parameter $s = 1$, then the probability that a data object is assigned to a single role (belongs to rank $\alpha = 1$) doubles the probability that the same data object is assigned to two roles (belongs to rank $\alpha = 2$). As value of $s$ increases, the number of data objects exclusively assigned to an individual role becomes larger as shown in Figure 4. Note, the value of $s$ should be greater than or equal to 1.

The Zipfian distribution can be used to generate heterogeneous RBAC-based workload in two steps as proposed in Algorithm 1. In the first step, data objects are classified into $n$ buckets where each bucket represents the number of total data objects assigned to a lattice level of Figure 3b. For example, data objects in bucket 1 are exclusively accessed by only one role. On the other hand, data objects in bucket $n$ are shared by all roles. The number of data objects in each bucket follows Zipfian distribution. In the second step, we assign data objects of bucket $i$ to randomly selected partitions at level $i$ of the lattice. Note, the number of partitions at level $i$ is $\binom{n}{i}$. 

![Diagram of RBAC](image-url)
5 SENSITIVITY OF DATACENTER USING SPECTRAL MODEL AND RISK MANAGEMENT

Based on the statistical property of access control workload, we propose the notion of data sensitivity based classification of cloud datacenters. The sensitivity classification is dependent on the level of sharing of data objects among the roles. In particular, we define the sensitivity of a datacenter as the average degree of sharing among its data objects. In case, the average degree of sharing is low, we term this datacenter to have a low sensitivity. The medium sensitivity class falls in the middle. The goal is to minimize the total risk in a highly sensitive datacenter environment. It can be noticed that the sharing of data objects and the classification of a datacenter can be modeled using the Zipfian distribution. The key parameter to characterize the sensitivity of datacenter is the scalar parameter \( s \) of the Zipfian density function shown in Equation 3. As shown in Figure 4, for a smaller value of \( s \), more data objects are uniformly distributed in the \( W \) vector of the spectral model of RBAC. In the following example, we illustrate how the Zipfian parameter \( s \) can be used to classify the sensitivity of a datacenter.

**Example 5.1.** For a datacenter with \( 0.5 \times 10^6 \) data objects, suppose we have three RBAC policies \( (P_1, P_2, P_3) \) each with \( n = 30 \) roles. Figure 4 shows a histogram of data objects across the spectral lattice. Depending on the Zipfian distribution, the three classes of datacenters, namely; High Sensitivity Datacenter (HSD), Medium Sensitivity Datacenter (MSD), and Low Sensitivity Datacenter (LSD), can be identified with respect to policies \( P_1, P_2, \) and \( P_3 \). For example, HSD has a large value of \( s \) \((s \geq 2)\) since the sharing of data objects among \( P_1 \) roles is very small. On the other hand, LSD has a small value of \( s \) \((1.5 > s \geq 1)\) depicting the case of extensive sharing of data objects among roles of policy \( P_3 \). The MSD has a value of \( s \) which falls in the middle \((2 > s \geq 1.5)\).

---

6 INTEGRATED SERVICE-VM VULNERABILITY MODEL

In order to quantify the probability of data leakage associated with intra- and inter-ISVM vulnerabilities, we use two models namely, Weakest Link model [32] and Dependency model [16] based on Common Vulnerability Scoring System (CVSS) [27].

6.1 Weakest Link Model

The Weakest Link model is to identify the vulnerability that requires the least amount of effort to exploit in order to obtain the desired level of privilege resulting in data leakage to the attacker. Selecting the Weakest Link involves choosing the single most critical vulnerability (as designated by the CVSS Base Score) of every system [32]. If there are several vulnerabilities with the same Base Score but different Impact Score or Exploitability Score, an expert should be present to judge their severity. It is not known whether the ease of exploitation (CVSS Exploitability Score), the effect of an attack (CVSS Impact Score), or a combination of the two (CVSS Base Score) is the most valuable to an attacker.

6.2 Dependency Model

The Weakest Link approach ignores casual and dependency relationships smog vulnerabilities. Such an approach may cause useful semantics of individual scores to be lost. The dependency relationship of vulnerabilities being ignored or handled in an arbitrary way brings doubts to the metric results and prevents their adoption. Moreover, only the attack probability is considered for aggregating CVSS scores in the Weakest Link approach which may limit the scope of the application and may give misleading results since different aspects demand different algebra for aggregating the scores.

This issue can be addressed by interpreting and aggregating CVSS scores from three aspects namely, probability, effort, and skill [16].

**Definition 6.1.** (The Cloud ISVM Model). Given the set of \( l \) services \( S = \{s_1, \ldots, s_l\} \) hosted in the cloud and \( V = \{v_1, v_2, \ldots, v_m\} \) as a set of \( m \) virtual machines available to VRM, let \( d_{i,j,k,q} \) represent the probability of data leakage among ISVM bi-tuple pairs \( (s_i, v_j) \) and \( (s_k, v_q) \). This probability is estimated by the resource vulnerability estimation component as shown in Figure 1a. The \( d_{i,j,k,q} \) can be computed based on Weakest Link model or dependency model. Accordingly, in order to
compute all inter- and intra-ISVM data leakage probabilities due to inter- and intra-ISVM vulnerabilities, the ISVM leakage probabilities can be modeled as a fully connected undirected graph $G(Q, E)$ where vertices in $Q = \{ (s_i, v_j) : s_i \in S \text{ and } v_j \in V \}$ represent ISVM and the weights on the edges represent $d_{i,j,k,q}$.

Risk functions: $\text{Risk}(w_{12}) = w_{12} \times \max(1 - d_{1311}, 1 - d_{1322})$

![Diagram of Risk calculation](image)

Fig. 5: Examples of risk probability in max and parallel models.

## 7 Resource Assignment

Based on $G(Q, E)$ graph, two models namely, max and parallel, can be used to estimate the risk of data leakage as illustrated in Figure 5a and 5b. Accordingly, we propose two cost functions namely RASP-MAX and RASP-PAR. We use both of these models and do not prefer one over the other as they provide independent cost functions.

### 7.1 Risk-aware Scheduling Problem - MAX (RASP-MAX)

The Risk-aware Scheduling Problem (RASP-MAX) is constructed with the following information given:

- spectral representation $W$ of a RBAC policy
- a role to service requirement matrix $\Gamma_{r_i,s_j}$ depicting that role $r_i$ requires service $s_j$
- the compatibility matrix $C$, where $C_{s_i,v_j}$ indicates which service $s_i$ is compatible with VM $v_j$
- and the vulnerability matrix represented as $G(Q, E)$ providing the probabilities of data leakage in terms of $d_{i,j,k,q}$

Then RASP-MAX is to minimize the total risk $\text{Risk}$ of data leakage by allocating the virtual resource to all the required services assuming all roles are active such that each virtual machine can host only one compatible service at a time. Formally, the optimization problem is:

Minimize $\text{Risk} = \sum_{w_p \in W} \text{Risk}(w_p)$

$$= \sum_{w_p \in W} w_p \times \sum_{r_a \in R} \text{Threat}(p, r_a) \times \max_B$$

$$\text{Where: A : } \forall s_i, s_k \in S, \forall v_j, v_q \in V, \forall r_b \in p$$

$$B : d_{i,j,k,q} \times I_{r_a,s_i,v_j} \times I_{r_b,s_k,v_q}$$

Such that:

1) $I_{r_a,s_i,v_j} \times I_{r_b,s_k,v_q} = 0$, when $s_i \neq s_k$
2) $\sum_{v_j \in V} I_{r_a,s_i,v_j} = 1$, when $\Gamma_{r_a,s_i} = 1$
3) $I_{r_a,s_i,v_j} = 0$, when $C_{s_i,v_j} = 0 \forall r_a \in R$

Where:

- $R, V, S$, and $d$ as given in Table 1 in Appendix
- $w_p$: number of tuples accessed by roles in set $p$ as defined in Equation 2

$$I_{r_a,s_i,v_j} = \begin{cases} 1 & \text{if } r_a \text{ is assigned to ISVM } (s_i,v_j) \\ 0 & \text{Otherwise} \end{cases}$$

$$\Gamma_{r_a,s_i} = \begin{cases} 1 & \text{if } r_a \text{ requires service } s_i \\ 0 & \text{Otherwise} \end{cases}$$

$$C_{s_i,v_j} = \begin{cases} 1 & \text{if service } s_i \text{ can run on virtual machine } v_j \\ 0 & \text{Otherwise} \end{cases}$$

$$\text{Threat}(p, r_a) = \begin{cases} 1 & \text{if } r_a \notin p \\ 0 & \text{Otherwise} \end{cases}$$ (see Equation 1)

### Theorem 7.1. RASP-MAX problem is NP-complete.

### 7.2 Risk-aware Scheduling Problem - PAR (RASP-PAR)

The Risk-aware Scheduling Problem (RASP-PAR) is constructed as follows. Given spectral representation $W$, the requirement matrix $\Gamma$, the compatibility matrix $C$, and the vulnerability matrix represented as $G(Q, E)$, then RASP-PAR is to minimize the total risk $\text{Risk}$ of data leakage by allocating the virtual resource to all the required services.

Minimize $\text{Risk} = \sum_{w_p \in W} \text{Risk}(w_p)$

$$= \sum_{w_p \in W} w_p \times \sum_{r_a \in R} \text{Threat}(p, r_a) \times \prod_{A}$$

$$A : \forall s_i, s_k \in S, \forall v_j, v_q \in V, \forall r_b \in p$$

$$B : (1 - d_{i,j,k,q}) \times I_{r_a,s_i,v_j} \times I_{r_b,s_k,v_q}$$

All the constraints of Equation 4 are also applied here. It can be noted that the notion of attackability is also captured in the two summations shown in Equation 5.

### Theorem 7.2. RASP-PAR problem is NP-complete.
8 Assignment Heuristics

In this section, we propose three heuristics for solving the RASP which can be deployed by the resource assignment component of VRM as shown in Figure 1a. The first heuristic is called the Best Fit Heuristic (BFH) and uses the best fit strategy. In BFH, each role is assigned to the best available virtual machine such that its probability of leakage and any increase in the total risk is kept to minimum. The second heuristic is called Single Move Heuristic (SMH) which recursively improves the assignment solution based on Initial Assignment (IA) till no further solution is possible (may not be optimal). For further refinement to SMH, we propose Multi Move Heuristic (MMH).

Since all the three heuristics use the spectral model and as $|W| = 2^n - 1$, in order to reduce the complexity of BFH, SMH and MMH, we propose an approximation strategy for workload characterization. The strategy is based on considering a smaller percentage of the total size of datacenter. Let such percentage be denoted as $D$. In particular, $D$ identifies the cutoff level $k$ in the lattice of $P(R)$ which can be used by BFH, SMH and MMH. The following lemma describes the process of determining this cutoff $k$.

**Lemma 8.1.** Given $W$ (spectral representation of RBAC), $s$ (the scalar parameter of the Zipfian distribution), and $D$ (the percentage of data considered), the cutoff level $k$ in the lattice of $P(R)$ is defined as $k = \min_{n \leq k} \frac{D}{100} \times H_{k,s}$ where $H_{k,s} = \sum_{i=1}^{k} i^{-s}$ is $k$th generalized Harmonic number.

For a given value $D$ of a datacenter, the spectral vector $W$ needs to be truncated accordingly as detailed in Lemma 8.1. The truncated vector $W'$ is defined as follows.

**Definition 8.1.** We define $W'$ to consist of all the partitions of $W$ starting from level one up to and including the cutoff level $k$ in the lattice of Figure 3b. The vector $W'$ is formally defined as: $W' = \{w_p | (w_p \in W) \land |p| \leq k\}$ where the size of $W'$ can be approximated in term of cutoff $k$, as giving by the following lemma.

**Lemma 8.2.** $|W'| \leq (n + 1)^k$

Note, different sensitivity classes of datacenter yield different cutoff levels for the same percentage $D$. Accordingly, the size of $W'$ varies. The example below illustrates how $D$ and the sensitivity classes can affect the value of $k$.

**Example 8.1.** For $W$ of Example 5.1, when $D = 70\%$, the cutoff ($k$) is 2 for HSD and is 8 for LSD. For $D = 95\%$ the cutoff for MSD is 18 as compared to 9 and 24 for HSD and LSD, respectively.

In following section, we describe BFH, SMH and MMH using a two-phase approach. In phase one, a heuristic uses a greedy approach to generate an assignment based on partitions in $W'$. In phase two, all the remaining partitions $(W - W')$ are assigned automatically with their associated roles. In Section 9, we provide a trade off between the accuracy of risk assessment based on the total workload represented by $W$ vs. approximate workload represented by $W'$.

---

**Algorithm 2: Best Fit Heuristic**

**Input:** Spectral representation of RBAC $W'$, vulnerability graph $G(Q,E)$, set of roles $R$, set of services $S$, requirement matrix $\Gamma$, compatibility matrix $C$, and initial assignment matrix $IA$.

**Output:** A new assignment matrix $I$ of roles to service and VM pairs.

1. $I = IA$;
2. Let $A = \{(r_a,s_i)\} \forall r_a \in R, \forall s_i \in S$ where $\Gamma_{r_a,s_i} = 1$;
3. while $A \neq \emptyset$ do
   4. Randomly select $(r_a,s_i)$ pair from $A$;
   5. Let $vList = \{v_q\} \forall \Gamma_{r_a,s_i} = 1$ for all $v_q \in V$;
   6. foreach $v_q \in vList$ do
      7. if $\forall r_a \in R, \forall s_j \neq s_i$ then
         8. remove $v_q$ from vList;
   9. Compute function RISK($r_a,s_i,v_q$) for all $v_q \in vList$;
10. let $v_q$ be the VM with minimum risk;
11. if $I_{r_a,s_i,v_q} == 0$ then
12. $I_{r_a,s_i,v_q} = 1$;
13. $A = A - (r_a,s_i)$;
14. return $I$;
15. Function RISK($r_a,s_i,v_q$) let $t = 0$;
16. foreach $v_b \in W'$ do
17. $w_p \in W'$ do
   18. construct $vul[r_a][r_b]$ based on assignment of $r_a$ to $(s_i,v_q)$;
19. $w_p \in W'$ do
20. $r_b \notin p$ do
   21. let $max = 0$;
   22. foreach $v_m \in p$ do
   23. if $vul[r_b][r_a] \geq max$ then
   24. $max = vul[r_b][r_a]$;
25. let $t = t + w_b \times max$
26. return $t$;

---

8.1 Initial Assignment (IA)

Initial assignment is a random assignment of services to VMs in order to find a feasible solution. IA is generated as follows: Given compatibility matrix $C$, IA represents $C$ as a bipartite graph with two sets of vertices namely $S$ (services) and $V$ (VM’s) such that there is and edge from service $s_i$ to virtual machine $v_j$ when $C_{s_i,v_j} = 1$. IA finds a maximum bipartite matching. The problem has a feasible solution if the maximum bipartite matching cardinality equals $|S|$. From requirement matrix $\Gamma$, if a role $r_a$ is requiring service $s_i$ then $IA_{r_a,s_i,v_j} = 1$ such that $(s_i,v_j)$ pair is the one produced by the maximum bipartite matching. This assignment guarantees a feasible solution however it may incur high risk.

8.2 Best Fit Heuristic (BFH)

The algorithm for BFH is presented in Algorithm 2. Given the initial assignment $IA$, BFH randomly selects a role and service pair $(r_a,s_i)$ from the requirement matrix $\Gamma$ and finds a new assignment with maximum reduction in resulting risk. We now explain the operation of the Algorithm as follows. The initial assignment matrix $IA$ is assigned to the new assignment matrix $I$ (Line 1). In Line 2, the algorithm constructs a set $A$ which represents elements of $\Gamma$ for which $\Gamma_{r_a,s_i} = 1$. A random $(r_a,s_i)$ pair is selected from set $A$ (Line 4). A list of VMs, named $vList$, is then constructed.
Algorithm 3: Single Move Heuristic


Output: A new assignment matrix $I$ of roles to service and VM pairs

1. flag = 1;
2. $I = IA$;
3. while flag = 1 do
4.   Let $A = \{(r_a,s_i)\}$ $\forall r_a \in R, \forall s_i \in S$ where $\Gamma_{r_a,s_i} = 1$;
5.   flag = 0;
6.   while $A \neq \phi$ and flag = 0 do
7.     Randomly select $(r_a,s_i)$ pair from $A$;
8.     Let vList = $\{v_q\}$ where $C_{s_i,v_q} = 1$ for all $v_q \in V$;
9.     foreach $v_q \in$ vList do
10.    if $\forall r_a \in R$, $I_{r_a,s_j,v_q} = 1$ $\land s_j \neq s_i$ then
11.       remove $v_q$ from vList;
12.    Compute function $RISK(r_a,s_i,v_q)$ for all $v_q \in$ vList;
13.    $\hat{v_q}$ be the VM with minimum risk;
14.    if $I_{r_a,s_i,\hat{v_q}} == 0$ then
15.       flag = 1;
16.       $I_{r_a,s_i,\hat{v_q}} = 1$;
17.    else
18.       $A = A - (r_a,s_i)$;
19. return $I$;

with all VMs that are compatible with the service $s_i$ excluding the VMs that have already been assigned to a different service (Lines 5-8). For each VM in the list, the risk is computed (Line 9). The VM with maximum reduction in risk, $\hat{v_q}$, is selected (Line 10). If the $\hat{v_q}$ is not in the current assignment, it is added to the current assignment. The selected pair $(r_a,s_i)$ is removed from set $A$ in Line 13 and the loop in Lines 3-13 iterates until the set $A$ is empty.

In order to find the maximum reduction in risk, BFH calls function $RISK(r_a,s_i,v_q)$ which calculates the total risk based on the assignment $I_{r_a,s_i,v_q} = 1$. In Lines 17 and 18 the function generates pairwise vulnerability between role $r_a$ and all other roles. The function then iterates through all $w_p \in W'$ and for each $w_p$ computes the risk based on the maximum vulnerability (in case of RASP-MAX) between $r_a \in p$ and $v_q \notin p$ in Lines 19-25. It returns the summation of product of $w_p$ and the maximum vulnerabilities based on assignment $I_{r_a,s_i,v_q} = 1$.

Lemma 8.3. The complexity of BFH is $O(m \times l \times n^3 \times |W'|)$.

Note, due to its linear complexity in terms of number of services and number of VMs, BFH provides a scalable solution for both RASP-MAX and RASP-PAR problems.

8.3 Single Move Heuristic (SMH)

The algorithm for SMH is presented in Algorithm 3. We now explain the operation of this algorithm as follows. The initial assignment matrix $IA$ is assigned to the new assignment matrix $I$ (Line 2). In Line 4, the initial members of $A$ represent elements of $\Gamma$ for which $\Gamma_{r_a,s_i} = 1$. A random ($r_a,s_i$) pair is selected from set $A$ (Line 7). A list of VMs is then constructed with all VMs that are compatible with the service $s_i$ excluding the VMs that have already been assigned to a different service (Lines 8-11). For each VM in the list, the risk is computed (Line 12). The VM with minimum risk is selected (Line 13). If the selected VM is not in the current assignment, it is added to the current assignment and the algorithm returns to Line 4. Otherwise, the random pair $(r_a,s_i)$ is removed from $A$ and algorithm returns to Line 7. The algorithm iterates in the inner loop Lines 6-18 until either the it finds a better assignment (flag = 1) or the set $A$ is empty which means no further improvement can be achieved for any $(r_a,s_i)$ in $A$.

Lemma 8.4. The complexity of SMH is $O(m \times l \times n^4 \times |W'|)$.

8.4 Multi Move Heuristic (MMH)

The algorithm for MMH is presented in Algorithm 4. We now explain the operation of the Algorithm as follows. The initial assignment matrix $IA$ is assigned to the new assignment matrix $I$ (Line 2). $IA$ is constructed as explained earlier. The $f(v_q)$ function returns the services $s_i$ hosted by $v_q$ based on the current assignment matrix (Line 5). The algorithm iterates for each possible pair of VMs ($v_i,v_j$). If both $v_i$ and $v_j$ host the same service ($f(v_i) = f(v_j)$), all roles assigned to these two VMs are determined (Line 9). The risk is then computed for each role belonging to the two VMs (Line 10). For a given role, if the risk is less for $v_i$, the role is assigned to $v_i$. Otherwise, the role is assigned to $v_j$ (Lines 12-16). If the new assignment different than the current assignment, it implies that the algorithm finds an assignment that reduces the risk and updates the new assignment to become the current assignment. Otherwise, the current assignment does not change (Lines 17-19). If $f(v_i) \neq f(v_j)$, the algorithm swaps the services if each VM is compatible with other VM’s service (Lines 21). The swap routine generates a new assignment $I$ and the algorithm updates the current assignment to the new assignment only if there is a reduction in the risk (Lines 23-34). The above procedure is repeated for all possible pairs of VMs. The algorithm iterates outer loop (Lines 2-24) until no further risk improvement can be achieved in the current assignment for all possible pairs of VMs.

Lemma 8.5. The complexity of MMH is $O(m^2 \times l \times n^4 \times |W'|)$.

8.5 Performance Metrics for Resource Assignment

We introduce two sets of metrics to compare the performance of the proposed BFH, SMH and MMH. The first set is concerned with the overall risk exhibited by all roles. The second set of metrics focuses on the risk incurred per role. The formal definitions of these metrics are presented in this subsection and the experiment results are given in the next section.

The data leakage risk is the main metric that needs to be minimized. The risk metric has two sub-metrics which we use to evaluate the proposed heuristics. The first risk metric is computed from the total workload which represents all the partitions in $W$. This risk metric represents the total risk and is denoted as $T$. In other words $T$ represents the potential risk for the entire datacenter. The second

This error is the second performance metric and it is a result of the heuristic’s lack of knowledge of all the partitions due to workload approximation by $W$. Formally, the risk metrics are given as:

$$T = \sum_{w_p \in W} Risk(w_p)$$  \hspace{1cm} (6)

$$P = \sum_{w_p \in W} Risk(w_p)$$  \hspace{1cm} (7)

Note, the difference between total risk and partial risk represents the relative risk error (δ). Formally, we define δ as:

$$\delta = AT - P$$, where $AT = \sum_{w_p \in W} AT(w_p)$  \hspace{1cm} (9)

Note, δ provides a quality measures of a heuristic in terms of its partial risk $P$ as compared to the maximum incurred risk (Attackability). A high value of δ implies that the resulting $P$ is far from the maximum risk, thus the heuristic performs well.

The second set of metrics are concerned with performance of heuristics at the role level. One such metric is the Discriminator Index (DI) which is an indirect indicator of performance of heuristic in term of risk reduction per role. Semantically, DI not only implies the role quality of reduction but also disproportionality in terms of risk management i.e. a heterogeneous bias in managing the risk. Formally, it is defined as following:

**Definition 8.2.** Given a scheduling heuristic, DI for a heuristic can be formally defined by extending the definition of generic discriminator index in [33]. For this purpose, given a role $r_i$, we define the role attackability ($AT_i$), the role partial risk ($P_i$), and the role quality of risk reduction $\delta_i$. Let:

$$AT_i = \sum_{w_p \in W' \cap r_i \cap P} AT(w_p)$$

$$P_i = \sum_{w_p \in W' \cap r_i \cap P} Risk(w_p)$$

$$\delta_i = AT_i - P_i$$

Accordingly, DI is defined as

$$DI = 1 - \frac{(\sum_{i=1}^{n} \delta_i)^2}{n \times \sum_{i=1}^{n} (\delta_i)^2}$$  \hspace{1cm} (10)

It can be noticed that $w_p$'s are “non linearly” dependent on parameter $s$ as given by Equation 3 and Algorithm 1 (Line 7-9). Accordingly both the cost functions given in Equations 4 and 5 are non-linearly dependent on $s$. Therefore any heuristic designed to solve these problems should reduce the overall risk “non-linearly” with the three classes of datacenters i.e. $HSD$, $MSD$ and $LSD$. As mentioned earlier, the Attackability (AT) parameter of these three classes namely $HSD$, $MSD$ and $LSD$ decreases in their respective order. Using this order, we can evaluate the quality of the solution of these heuristics based on their risk reduction and how sensitive is such reduction in terms of $s$, $D$, $n$ and $m$. In particular, the sensitivity analysis focuses on the number of roles and number of VMs.

## 9 Performance Evaluation

In this section, we present a detailed evaluation of results in terms of $s$ (the sensitivity parameter of the Zipfian distribution), $D$ and the size of the problem i.e. number of VMs ($m$), and roles ($n$). These are independent parameters which are critical to the risk management of the RASP-MAX and RASP-PAR problems. The results are organized in three categories based on classification of cloud datacenters i.e. $HSD$, $MSD$, and $LSD$ as mentioned in Section 5. For each category, we present the performance results of three heuristic algorithms for the metrics $P$, $E$ and $DI$. These results are presented for varying percentage $D$. 

---

Algorithm 4: Multi Move Heuristic

**Input:** Spectral representation of RBAC $W'$, vulnerability graph $G(Q, E)$, set of roles $R$, set of services $S$, requirement matrix $\Gamma$, compatibility matrix $C$, and initial assignment $IA$. 

**Output:** A new assignment matrix $I$ of roles to service and VM pairs

1. flag = 1;
2. $I = IA$;
3. while flag = 1 do
4. flag=F;
5. Let $f(v_i) = v_i$ when $I_{r_a, v_i}, v_q = 1$;
6. foreach $v_i \in V$ do
7. foreach $v_j \in V$ do
8. if $f(v_i) = f(v_j)$ then
9. Let $F = \{r_a\}$ when $I_{r_a, f(v_i), v_q} = 1$; where $v_q \in \{v_i, v_j\}$;
10. Compute $g_{r_a} = Risk(r_a, f(v_i), v_q)$ for all $r_a \in F$;
11. Let $\hat{I} = I$;
12. foreach $r_p \in F$ do
13. if $g_{r_p} \leq g_{r_a}$ then
14. $I_{r_a, f(v_i), v_q} = 1$
15. else
16. $I_{r_p, f(v_i), v_q} = 1$
17. if $\hat{I} \neq I$ then
18. flag = 1;
19. $I = \hat{I}$;
20. else
21. if $C_{f(v_i), v_q} = C_{f(v_j), v_q}$ then
22. Let $\hat{I} = I$;
23. if swap($v_i, v_j$) reduces total risk then
24. $I = \hat{I}$;
25. flag = 1;
26. return $I$;

---

sub-metric of risk is based on all the partitions of $W'$ which is used to study the performance of the heuristics and compare their effectiveness. This sub-metric is called partial risk $P$ and corresponds to the risk resulting from the workload approximation by $W'$. Formally, the risk metrics are given as:

$$T = \sum_{w_p \in W} Risk(w_p)$$  \hspace{1cm} (6)

$$P = \sum_{w_p \in W} Risk(w_p)$$  \hspace{1cm} (7)

As mentioned in Definition 3.2, the Attackability $AT$ measures the maximum amount of data leakage risk for a given policy. The difference between the Attackability of a policy and the resulting partial risk represents the quality of risk reduction ($\delta$). Formally, we define $\delta$ as:

$$\delta = AT - P$$, where $AT = \sum_{w_p \in W'} AT(w_p)$  \hspace{1cm} (9)

---

2168-7161 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Fig. 6: Partial Risk $P$ with a problem size of 70 roles and 50 VMs for RASP-MAX.

Fig. 7: Error $E$ with a problem size of 70 roles and 50 VMs for RASP-MAX.

Fig. 8: Discrimination Index $DI$ with a problem size of 70 roles and 50 VMs for RASP-MAX.

Fig. 9: Partial Risk $P$ with a problem size of 120 roles and 200 VMs for RASP-MAX.

Fig. 10: Error $E$ with a problem size of 120 roles and 200 VMs for RASP-MAX.
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2015.2453981, IEEE Transactions on Cloud Computing

9.1 Experimental Setup

In our evaluation, we use three types of cloud datacenters i.e. LSD, MSD, and LSD as mentioned in Section 5. In addition, we vary the percentage of datacenter $D$ from 70% to 95% where the complexity of the problem increases with $D$. To generate ISVM matrix values, we assume that probability of data leakage among VMs belonging to different IaaS providers is negligible. We generate four classes of inter-ISVM vulnerabilities representing four IaaS providers security control [5]. In addition, the intra-ISVM vulnerabilities are randomly generated from four security groups, named highly secure, medium secure, low secure, and insecure.

All intra-ISVM vulnerabilities are higher than inter-ISVM vulnerabilities due to the fact that in general, virtualization isolation is more secure than application/VM isolation. A uniformly random IA used in terms of bipartite graph representing the VM-services compatibility matrix is generated such that each service is compatible with at least one VM and each VM is compatible with at least one service. Also, the role-service requirement is randomly generated using a uniform distribution such that each role requires at least one service and each service is required by one or more roles. In this experiment, we have two different problem size settings in terms of number of roles and number of VMs. The first problem size assumes 70 roles and 50 VMs and the
second problem size, which is the larger scale problem size, assumes 120 roles and 200 VMs. The number of services is 20 in both cases.

9.2 Results for RASP-MAX
In this section, we discuss the aforementioned performance metrics i.e. Partial Risk ($P$), Relative Risk Error ($E$), and Discrimination index ($DI$).

9.2.1 Partial Risk ($P$)
For the initial assignment (IA), we observe $P$ increases as $s$ increases from a low value to a high value corresponding to LSD and HSD cases respectively as shown in Figure 6. The reason being that Attackability ($AT_i$) as per Definition 3.4 and Algorithm 1 (Line 7-9) increases with $s$ for a given $D$. Since the initial assignment is a random assignment and it only ensures feasibility, it follows the attackability curve. However, among the three heuristics in general, SMH outperforms BFH since it iterates several times until obtaining a local minimum in comparison to BFH that iterates only once. MMH always outperforms other heuristics since it can change multiple assignments simultaneously for each iteration in comparison to SMH which changes only one assignment and hence as expected, MMH performs better. We observe that $P$ is higher for LSD than HSD. Also, $P$ converges with increasing $s$ and $D$. Furthermore, the following observations of $P$ and $AT$ from Figure 6 are explained in terms of the quality of reduction ($\delta$) as defined in Equation 9:

- For a given heuristic, $\delta$ increases with $D$. The reason being that with the increase in $D$, the number of partitions i.e. $w_i$s considered by the heuristics increases resulting in an improved assignment decision.
- For a given $D$, $\delta$ increases with $s$. The reason being that with increasing value of $s$, heterogeneity of partitions is more pronounced resulting in an improved value of $\delta$.

9.2.2 Relative Risk Error ($E$)
The metric $E$ elucidates the effect on assignment decisions when ($W − W'$) partitions of data are not considered by the proposed heuristics. We observe from Figure 7 that $E$ decreases as $D$ increases. The reason being that as more partitions are considered for assignment decision, the difference between $T$ and $P$ is less thus reducing $E$. In other words, excluding ($W − W'$) partitions of data result in worse assignment decisions and hence resulting in a high value of $E$. In addition, we can notice for a given $D$, $E$ increases with increasing $s$. As mentioned above, $AT$ is higher for HSD as compared to LSD. Hence, if ($W − W'$) partitions are not considered by the heuristic for the case of HSD, the resulting $E$ is higher in comparison to LSD for the same partitions.

9.2.3 Discrimination Index ($DI$)
The effect of $D$ and $s$ on heuristics is further illustrated using the $DI$ metric. We observe from Figure 8 that $DI$ increases as $s$ increases from low to high corresponding to LSD and HSD cases respectively. The reason being that $DI$ as mentioned earlier can be viewed as the degree of heterogeneity among the quality of risk reduction ($\delta$, $s$) associated with role $r_i$. For the case of HSD, the heterogeneity of roles attackability ($AT_i$) is higher than the case for LSD due to high value of $s$ for HSD. We expect $\delta$ heterogeneity to be higher in HSD than LSD. According to Equation 10, this leads to a higher value of $DI$ for HSD compared to LSD. We also observe that $DI$ decreases as $D$ increases from low to high i.e. from 70% to 95%. Note, for different percentage $D$, the cutoff parameter $k$ in Lemma 8.1 is less in lower $D$ as compared to higher $D$. Therefore, as $k$ increases, the heterogeneity of the RBAC policy in term of $w_i$s decreases. Since $DI$ represents such heterogeneity, $DI$ for a low percentage $D$ is higher as compared to high percentage $D$. For MSD ($s = 1.5$), an important observation (as shown in Figure 8) is that there is a rapid shift in $DI$ parameter form $D = 70\%$ to $D = 95\%$. The figure displays the interplay of parameter $DI$ and $D$ for a medium value of $s$. In essence, for small value of $D$, the heterogeneity of the RBAC policy is similar between MSD and HSD. As $D$ increases, the MSD heterogeneity is in the middle of LSD and HSD.

Same observations as mentioned above with respect to $P$, $DI$ and $E$ parameters can be made while increasing the problem size in terms of number of roles and VMs. For completeness purpose, we provide Figures 9, 10, and 11 that depict the results for large scale problems. For large problems, we observe that the overall difference in $P$ appears to diminish and the performance of heuristics converges. This is because while increasing the number of VMs, there are more options for assignment and hence all the heuristics tend to perform almost equally good. However, for a small number of VMs i.e. for a smaller problem size, the heuristic with a higher intelligence and high complexity yields an improved performance as is the case with MMH. Hence, we conclude that for a large problem size, BFH is a preferable choice since it is; scalable (Lemma 8.3), performs equally good as other heuristics in terms of $P$ (Figure 9), and its resulting $E$ for a low value of $D$ is also low (Figure 10).

9.3 Results for RASP-PAR
Regarding the performance of the three heuristics, we observe similar phenomenons with RASP-PAR as observed with RASP-MAX with respect to $P$, $DI$ and $E$ parameters. For example MMH consistently outperforms other heuristics. Note, the value of $E$ for BFH case is lower than the values of the other two heuristics. The performance results for RASP-PAR are shown in Figures 12, 13, and 14. It can be noted that although no cross comparison between RASP-MAX and RASP-PAR is intended since they represent cost functions of two different models of ISVM vulnerability, RASP-PAR produces higher risk than RASP-MAX strictly by definition of ISVM and not by comparison. By definition, due to highly parallel terms to compute vulnerability, it results in a high value of $P$ as depicted in Figure 12. For larger problems, even for RASP-PAR, BFH is a better choice for the three reasons given for the case of RASP-MAX.

10 Conclusion
The security of datacenters is becoming a primary issue in deploying cloud cloud computing . In particular, the virtualization and multitenancy features of cloud computing
that allow sharing of resources among potentially untrusted tenants exacerbate the security challenges in terms of increased risk of data leakage. In this paper we have presented a formal model for RBAC policy which was casted for formulating a risk aware VM assignment problem. We propose the notion of sensitivity in datacenters with the objective of minimizing the risk of data leakage. We present three assignment heuristics and compare their relative performance.

Acknowledgement: This work was partially supported by NSF grant IIS-0964639.

References


