

Security-Aware Resource Allocation Framework for Cloud Computing Environments

Shoeb Ali^{1*}, Adarsh Raj¹, Jatin¹, Krishna Bhatia¹, Parth Tiwari¹, Ramandeep Kaur¹

¹Department of Computer Science and Engineering, Chandigarh University, B1 Block (Nalanda), Mohali-140413, Punjab, India.

Email: shoebali2086@gmail.com

ARTICLE INFO

Article history:

Received 18 Sep 2026
Accepted 23 Sep 2026
Available online 30 Sep 2026

Keywords:

Cloud computing; Cloud security;
Security-aware resource
allocation; Virtual machine
placement; Dynamic threat
detection; CloudSim

Indexed in:



and in major libraries

ABSTRACT

Most cloud schedulers determine workload placement primarily by assessing whether candidate hosts have sufficient CPU, memory, and bandwidth. Security factors, such as patch status, vulnerability exposure, co-resident threats, and historical host behavior, are rarely integrated directly into resource allocation decisions. When considered, security is often applied as a simple pass or fail filter alongside conventional capacity-based scheduling. This paper proposes the Security-Aware Resource Allocation Framework (SARAF), which incorporates security indicators directly into workload placement decisions. For each candidate host, SARAF calculates a composite Security Score based on three components: recency-weighted trust history, vulnerability exposure, and short-window dynamic threat signals. This score is combined with resource availability through a weighted Allocation Score that automatically adjusts according to the security sensitivity of the incoming workload. Placement is performed using a greedy algorithm with bounded lookahead to reduce the concentration of workloads on highly ranked hosts. SARAF was evaluated using CloudSim with 50 heterogeneous hosts and 300 virtual machine requests across three workload regimes and compared with FCFS, Round Robin, Traditional Load Balancing, and Trust-Based Allocation. Under moderate workload conditions, SARAF reduced high-risk placements by approximately 61% compared with Round Robin and 38% compared with Trust-Based Allocation, while maintaining resource utilization within a few percentage points of the best throughput-focused baseline. These security improvements involve measurable tradeoffs, including approximately fivefold higher placement latency than FCFS and increased rejection rates under bursty workloads. Overall, the results demonstrate that security-aware scheduling can substantially reduce risky workload placements while preserving competitive resource utilization.

© 2026 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see <https://creativecommons.org/licenses/by/4.0/>

Introduction

Cloud computing derives much of its economic and operational efficiency from the sharing of physical infrastructure among multiple tenants, with resource schedulers determining how workloads are allocated across available hosts. A substantial body of cloud scheduling research has therefore focused on improving allocation efficiency through reduced makespan, improved load balancing, lower energy consumption, and fewer service-level agreement (SLA) violations [1,2]. However, comparatively less attention has been given to the security implications of the placement decision itself. In many cloud environments, security mechanisms such as network segmentation, hypervisor hardening, and intrusion detection operate primarily after

a virtual machine (VM) has already been assigned to a host. Consequently, host security posture may remain insufficiently integrated into the initial resource allocation decision [3].

This separation between resource allocation and security is increasingly difficult to justify in modern multi-tenant cloud environments. Co-residency threats, including cache-timing and memory-bus side-channel attacks, can potentially allow one VM to infer information about neighboring workloads. Compromised tenant accounts may also facilitate lateral movement, while misuse of privileged host access represents an additional security concern. Importantly, these risks are not captured by conventional resource-availability metrics. A host may have sufficient CPU, memory, and bandwidth while

remaining unsuitable for a security-sensitive workload because of outdated software, unpatched vulnerabilities, a history of anomalous behavior, or emerging threat indicators. Traditional approaches such as First-Come, First-Served (FCFS), Round Robin, and resource-oriented dynamic load balancing generally do not incorporate these security signals directly into placement decisions.

This study argues that resource availability and host security posture should be incorporated into a unified scheduling objective rather than evaluated as independent considerations. We refer to this approach as security-aware resource allocation and distinguish it from narrower strategies such as trust-aware scheduling, which primarily relies on historical host behavior, and risk-aware scheduling, which often represents security risk through a limited set of aggregated indicators [4]. To address these limitations, we propose the Security-Aware Resource Allocation Framework (SARAF). SARAF integrates trust history, vulnerability exposure, and dynamic threat information into a composite Security Score. This score is subsequently combined with resource availability through a tunable Allocation Score, allowing the relative emphasis on security and performance to vary according to workload sensitivity. The main contributions of this study are as follows:

1. **Composite Security Score:** A security assessment model that integrates three independently measurable components: trust history, vulnerability exposure, and dynamic threat detection. These components represent security signals that can potentially be obtained from existing cloud monitoring and security infrastructure.
2. **Workload-sensitive Allocation Score:** A unified formulation that combines the Security Score with resource availability and allows the security-performance trade-off to vary according to workload sensitivity rather than remaining fixed across all workloads.
3. **Security-aware placement algorithm:** A greedy VM placement algorithm with bounded lookahead that converts the Allocation Score into VM-to-host mappings while incorporating a secondary load-balancing consideration and maintaining reasonable computational complexity.
4. **Comparative CloudSim evaluation:** An evaluation of SARAF across three workload intensities against four baseline scheduling approaches: FCFS, Round Robin, Traditional Load Balancing, and Trust-Based Allocation. The evaluation considers both security-related and resource-performance metrics.
5. **Analysis of performance-security trade-offs:** A systematic assessment of the computational overhead, resource-utilization effects, and

request-rejection behavior associated with security-aware allocation, identifying both the benefits and practical limitations of the proposed framework.

The remainder of this paper is organized as follows. Section 2 reviews existing approaches to cloud resource allocation and security-aware scheduling, and Section 3 identifies the research gaps addressed by this study. Section 4 formulates the resource allocation problem. Sections 5 through 7 describe the proposed SARAF framework, its system architecture, and the associated mathematical models. Section 8 presents the placement algorithm, while Sections 9 through 11 describe the methodology, experimental setup, and evaluation metrics. Section 12 presents and discusses the results, and Section 13 compares SARAF with the baseline approaches. Finally, Sections 14 through 16 discuss the limitations, future research directions, and conclusions of the study.

2. Literature Review

2.1 Traditional Resource Allocation

Cloud resource scheduling has evolved from traditional operating-system process scheduling and grid-computing approaches. First-Come, First-Served (FCFS) and related priority-based methods allocate virtual machines (VMs) primarily according to arrival order and resource feasibility [5]. These approaches remain attractive because of their low computational complexity, predictability, and ease of implementation. However, FCFS does not explicitly optimize throughput, load distribution, energy efficiency, or security and may suffer from head-of-line blocking when large resource requests delay smaller requests that could otherwise be accommodated by available hosts. Bin-packing approaches, including First-Fit and Best-Fit, can improve resource utilization by increasing packing efficiency, but their placement decisions remain predominantly resource-oriented. Consequently, host security posture is generally not incorporated directly into the allocation objective.

2.2 Load Balancing Techniques

Load-balancing approaches such as Round Robin, Weighted Round Robin, Least Connections, and dynamic threshold-based methods aim to distribute workloads more evenly across available computing resources [7]. Round Robin provides a simple and computationally inexpensive mechanism for workload distribution but does not account adequately for heterogeneous host capacities or security conditions. Weighted variants partially address differences in host capacity by assigning allocation priorities according to predefined resource characteristics. Dynamic load-balancing methods further improve responsiveness by

considering current utilization and directing workloads toward less heavily loaded hosts [6].

Despite these improvements, conventional load-balancing strategies primarily optimize resource utilization and performance. A lightly loaded host may still present substantial security risks because of unpatched vulnerabilities, configuration weaknesses, previous anomalous activity, or emerging threat indicators [7]. Therefore, resource availability and host security cannot be assumed to represent equivalent indicators of placement suitability.

2.3 Trust-Based Scheduling

Trust-based scheduling introduces security-related information into resource allocation by assigning reputation or trust values to hosts, tenants, or other cloud entities [4,8]. Such scores may be derived from historical observations including SLA compliance, system availability, anomaly records, previous security incidents, and provider or peer feedback. Trust values are commonly calculated using weighted historical averages, although probabilistic and Bayesian approaches have also been proposed to account for uncertainty and incomplete observation histories [8].

These approaches represent an important advancement over purely resource-oriented scheduling because they recognize that hosts may differ in their security reliability. However, trust-based models are inherently dependent on historical observations. A host with a strong historical record may experience a newly emerging vulnerability or compromise before sufficient negative observations accumulate to substantially alter its trust score. Although recency weighting can increase sensitivity to recent events, historical trust alone cannot provide a fully responsive indication of rapidly changing security conditions. This limitation motivates the integration of more dynamic security indicators into allocation decisions [9].

2.4 Risk-Aware Scheduling

Risk-aware scheduling extends security-conscious resource allocation by incorporating estimates of potential security risk rather than relying exclusively on historical trust [10]. Such approaches may characterize risk according to the probability of a threat, its potential impact, vulnerability exposure, or combinations of related security factors. By incorporating prospective risk information, these methods can potentially identify unsuitable hosts before adverse events are reflected in historical trust records.

However, existing risk-aware approaches frequently represent security risk as an aggregated scalar derived from a relatively limited set of inputs [10]. Although useful for ranking candidate hosts, highly aggregated measures may reduce interpretability and make it difficult to determine which security factors contribute most strongly to a particular placement decision. Maintaining visibility of individual security components

can therefore improve transparency and auditability while still permitting their integration into a unified allocation objective.

2.5 Security-Aware VM Placement

Security-aware VM placement directly incorporates security considerations into the selection of physical hosts. Existing approaches have addressed concerns such as malicious co-residency, side-channel attacks, tenant isolation, and the distribution of workloads across hosts to limit the potential impact of host compromise [11]. These strategies demonstrate that placement itself can serve as an important security control rather than functioning solely as a resource-management operation. Nevertheless, many security-aware placement approaches focus on a specific threat model or implement security primarily as a hard eligibility constraint [3, 11]. Under such strategies, a candidate host either satisfies predefined security requirements or is excluded from consideration. Although effective in specific scenarios, binary security constraints provide limited flexibility for balancing security against resource availability and performance. A continuous security measure that incorporates multiple security indicators may provide a more adaptable basis for placement decisions, particularly when workloads differ in their security sensitivity.

2.6 AI-Based Resource Allocation

Machine learning and reinforcement learning have increasingly been applied to cloud resource allocation for workload prediction, demand forecasting, resource optimization, and adaptive VM placement [12]. These methods can capture complex relationships among host characteristics, workload demands, and system performance and can adapt allocation policies in response to changing cloud conditions.

However, many AI-based resource allocation approaches remain primarily oriented toward performance objectives such as latency, energy consumption, utilization, makespan, and operational cost [13]. When security is considered, it may be incorporated as a secondary constraint or penalty rather than as an explicit and interpretable component of the allocation objective. Furthermore, learned scheduling policies can introduce interpretability challenges, particularly in security-sensitive environments where operators may need to understand why a particular host was selected or avoided.

Taken together, the existing literature demonstrates substantial progress from resource-oriented scheduling toward trust-aware, risk-aware, security-aware, and intelligent allocation strategies. Nevertheless, important limitations remain in the integration of heterogeneous security signals, responsiveness to dynamic threats, interpretability of security assessments, and the ability to balance security and resource efficiency according to workload sensitivity. These limitations provide the basis

for the specific research gaps identified in the following section.

3. Research Gap Identification

Based on the preceding review of traditional, load-balancing, trust-based, risk-aware, security-aware, and AI-based resource allocation approaches, five key research gaps were identified. These gaps concern the integration of heterogeneous security information, responsiveness to dynamic threats, representation of security within placement decisions, adaptation to workload sensitivity, and comparative evaluation.

Gap 1: Fragmented integration of security signals.

Existing security-conscious resource allocation approaches commonly emphasize individual security dimensions, such as trust, vulnerability, or risk, rather than integrating multiple complementary indicators within a unified assessment. Consequently, a host may exhibit a favorable profile according to one security dimension while presenting substantial risk according to another. A systematic mechanism for combining historical trust, vulnerability exposure, and dynamic threat information within the placement decision remains insufficiently addressed [8,10, 11].

Gap 2: Limited responsiveness to dynamic security threats. Trust-based scheduling primarily relies on historical observations and therefore may respond slowly to newly emerging security events. A host with a strong historical trust record may experience a recent compromise or rapidly changing threat condition before sufficient negative observations accumulate to substantially alter its trust score. The integration of frequently updated threat intelligence and anomaly indicators into resource allocation decisions remains limited [4, 8].

Gap 3: Predominantly constraint-based representation of security. Many security-aware VM placement approaches represent security as a binary eligibility condition in which a host either satisfies predefined requirements or is excluded from consideration. Although such constraints can prevent placement on clearly unsuitable hosts, they provide limited flexibility for balancing security posture against resource availability and performance [3,10]. A continuous and interpretable security measure could enable more flexible placement decisions while maintaining explicit control over the security-performance trade-off.

Gap 4: Limited adaptation to workload-specific security requirements. Cloud workloads differ substantially in their security sensitivity and operational requirements. However, many existing allocation strategies apply fixed security priorities or require separate policy rules for different workload classes [12]. There remains a need for a unified and mathematically explicit allocation objective in which the relative emphasis on security and resource efficiency can be

adjusted according to workload sensitivity without requiring separate scheduling policies.

Gap 5: Limited comparative evaluation across security and performance dimensions. Security-aware, trust-based, and risk-aware scheduling approaches are often evaluated using different workloads, baseline algorithms, and performance metrics, making direct comparison difficult. Comprehensive evaluation against both conventional resource-oriented schedulers and security-conscious alternatives under consistent experimental conditions remains limited. Furthermore, security improvements should be assessed together with their associated effects on resource utilization, SLA violations, placement latency, load distribution, and request rejection.

To address these research gaps, we designed the proposed Security-Aware Resource Allocation Framework (SARAF). Specifically, SARAF integrates historical trust, vulnerability exposure, and dynamic threat information into a composite Security Score; combines this score with resource availability through a workload-sensitive Allocation Score; and evaluates the resulting placement strategy against multiple baseline approaches using both security and performance metrics [13]. The following section formally defines the resource allocation problem addressed by the proposed framework.

4. Problem Statement

Consider a cloud data center comprising a heterogeneous set of physical hosts

$$H = \{h_1, h_2, \dots, h_n\},$$

where each host is characterized by a resource-capacity vector and a set of security-relevant attributes. Let

$$V = \{v_1, v_2, \dots, v_m\}$$

denote a stream of virtual machine (VM) requests, where each VM is characterized by a resource-demand vector and an SLA-sensitivity class. The objective is to determine a placement function

$$f: V \rightarrow H$$

that assigns each accepted VM to an appropriate physical host while jointly considering resource feasibility, security posture, and computational efficiency.

Specifically, the placement function should satisfy the following objectives:

1. **Resource feasibility:** Ensure that the allocation of an accepted VM does not exceed the available CPU, memory, bandwidth, or other relevant resource capacities of the selected host.
2. **Security-aware placement:** Prefer hosts with stronger security posture, as represented by the Security Score, while accounting for the security sensitivity of individual workloads.
3. **Resource efficiency and load distribution:** Maintain effective utilization of available resources and balanced workload distribution while limiting the performance cost associated with security-aware placement.

4. **Computational efficiency:** Maintain sufficiently low per-decision computational overhead for online VM placement, with scalability approaching linear complexity with respect to the number of candidate hosts.

Conventional resource-oriented schedulers can be represented as optimizing a resource-performance objective $g(V, H)$, while security-related characteristics of candidate hosts are either omitted or considered separately. In contrast, SARAF integrates resource efficiency and security posture within a unified objective:

$$\Phi(V, H) = \alpha g(V, H) + \beta s(V, H),$$

where $g(V, H)$ represents the resource-performance utility of a placement, $s(V, H)$ represents its security utility, and α and β are tunable weighting parameters controlling the relative importance of resource performance and security, respectively. For a normalized weighted formulation,

$$\alpha + \beta = 1, \alpha, \beta \geq 0.$$

The objective is therefore to identify a feasible placement that approximately maximizes

$$\Phi(V, H)$$

subject to host resource-capacity constraints and acceptable computational overhead. Rather than solving the global placement problem exactly, SARAF employs an efficient heuristic placement strategy suitable for online scheduling. The subsequent sections define how the security and resource components are quantified and incorporated into individual VM-to-host placement decisions.

5. Proposed Security-Aware Resource Allocation Framework

The proposed Security-Aware Resource Allocation Framework (SARAF) integrates host security information directly into the virtual machine (VM) placement process rather than treating security as a separate post-allocation consideration. The framework represents host security posture through measurable and incrementally updated indicators and combines these indicators with resource availability to support transparent and security-aware placement decisions. SARAF consists of five principal components:

- **Security Score (SS):** A composite host-level security measure derived from Trust Score, Vulnerability Assessment, and Dynamic Threat Detection. The Security Score provides an integrated representation of the current security posture of each candidate host.
- **Trust Score (TS):** A historical measure of host reliability derived from previous behavioral observations, including SLA compliance, anomaly records, and security-related events. TS is maintained as an independently interpretable component because it represents

longer-term host behavior and changes more gradually than dynamic security indicators.

- **Resource Availability (RA):** A normalized measure of the computational resources currently available on each host, including CPU, memory, and bandwidth capacity. RA represents the resource-efficiency component of the placement decision.
- **Vulnerability Assessment (VA):** A periodically updated measure of host vulnerability exposure based on factors such as patch status, known vulnerabilities, exposed services or ports, and deviation from a hardened configuration baseline. Higher VA values represent greater vulnerability exposure.
- **Dynamic Threat Detection (DTD):** A frequently updated measure of short-term security conditions derived from indicators such as anomalous network activity, failed authentication attempts, and matches with threat-intelligence information. DTD is designed to capture emerging threats that may not yet be reflected in historical trust information.

These components contribute to two principal derived measures. First, the **Security Score** integrates TS, VA, and DTD to provide a composite assessment of host security posture, as defined in Section 7.1. Second, the **Allocation Score** combines the Security Score with Resource Availability, as described in Section 7.2. The Allocation Score is subsequently used by the placement algorithm to rank resource-feasible candidate hosts for each incoming VM request.

An important design characteristic of SARAF is the use of different update frequencies for its security components. Vulnerability-related information, including patch status and known vulnerability exposure, generally changes at a slower rate and can therefore be refreshed periodically without imposing unnecessary scanning overhead. In contrast, dynamic threat indicators may change rapidly and require more frequent updates to capture emerging security events. Accordingly, VA and DTD are maintained at different update frequencies, reflecting the distinct temporal characteristics of the information they represent.

This multi-timescale design enables SARAF to combine relatively stable vulnerability information and historical trust with rapidly changing threat indicators. Consequently, a host with a strong historical Trust Score can be reassessed when new vulnerability or threat information becomes available, reducing dependence on historical observations alone. This design directly addresses the limited responsiveness of trust-based scheduling identified in Gap 2 and provides the

foundation for the system architecture and scoring mechanisms described in the following sections.

6. System Architecture

SARAF is organized into five interconnected layers that support the flow of information from incoming VM requests to host selection and post-placement re-evaluation. The layered architecture separates functions according to their roles and update frequencies, enabling resource information, historical security data, vulnerability information, and dynamic threat indicators to be processed systematically within the allocation framework.

Layer 1: Request Intake and Classification. Incoming VM requests are characterized according to their resource requirements, including vCPU, memory, storage, and bandwidth, and assigned an SLA-sensitivity class: critical, standard, or best-effort. The sensitivity class may be specified by the tenant or determined from predefined workload characteristics. This classification determines the relative emphasis placed on security when the Allocation Score is calculated for the incoming request.

Layer 2: Security Intelligence Collector. This layer collects the security-related information required to calculate the Trust Score (TS), Vulnerability Assessment (VA), and Dynamic Threat Detection (DTD) measures. TS is supported by historical host-behavior information, including SLA violations, anomaly records, and previous security incidents. VA is derived from vulnerability-related information such as patch status and exposed services or ports. DTD incorporates frequently updated telemetry, including failed authentication activity, anomalous network behavior, and matches with threat-intelligence indicators. Together, these data sources provide historical, vulnerability-related, and dynamic security information for each host.

Layer 3: Scoring Engine. The Scoring Engine processes the security information collected in Layer 2 to calculate the Security Score for each host. It also obtains current resource-capacity information from the underlying infrastructure to determine Resource Availability. Security Score and Resource Availability are normalized to a common $[0, 1]$ scale, enabling their integration within the Allocation Score without differences in measurement units affecting the calculation.

Layer 4: Allocation Decision Engine. For each incoming VM request, the Allocation Decision Engine first identifies hosts that satisfy the VM's resource requirements. It then calculates the Allocation Score for each resource-feasible candidate host using the host's Security Score, Resource Availability, and the workload's SLA-sensitivity class. The candidate hosts are subsequently evaluated using the greedy placement

strategy with bounded lookahead described in Section 8, and the highest-ranked suitable host is selected for placement.

Layer 5: Feedback and Re-Evaluation Loop. Following placement, the framework monitors relevant outcomes, including SLA compliance and newly detected anomaly or security events. These observations are returned to the Security Intelligence Collector and incorporated into the host's security history. When a significant security event is detected, the Security Score of the affected host is recomputed without waiting for the next scheduled refresh cycle. This event-triggered re-evaluation enables subsequent placement decisions to reflect newly available security information.

Information therefore flows through SARAF as a continuous feedback process. An incoming VM request is classified in Layer 1, while security information is collected and maintained in Layer 2. Layer 3 transforms host security and resource information into normalized scores, and Layer 4 uses these scores together with workload sensitivity to determine the VM placement. Post-placement observations are then processed through Layer 5 and returned to Layer 2, allowing the security state of each host to be updated over time. This closed-loop architecture prevents host security posture from being treated as a static attribute and enables placement decisions to respond to changes in historical behavior, vulnerability exposure, and dynamic threat conditions.

7. Mathematical Model

7.1 Security Score Calculation

For each host h_i , the Security Score SS_i is calculated as a weighted combination of the Trust Score TS_i , Vulnerability Assessment VA_i , and Dynamic Threat Detection DTD_i . All three components are normalized to the interval $[0, 1]$. Because higher values of VA_i and DTD_i represent greater vulnerability exposure and threat activity, respectively, their complements are used so that all components of the Security Score follow a consistent interpretation in which higher values indicate a more secure host:

$$SS_i = w_t TS_i + w_v (1 - VA_i) + w_d (1 - DTD_i)$$

subject to

$$w_t + w_v + w_d = 1, w_t, w_v, w_d \geq 0.$$

Here, w_t , w_v , and w_d represent the relative contributions of historical trust, vulnerability exposure, and dynamic threat information, respectively. This formulation ensures that a host with a high Trust Score, low vulnerability exposure, and low dynamic threat activity receives a higher overall Security Score.

The Trust Score is calculated as a recency-weighted average of historical behavioral observations:

$$TS_i = \frac{\sum_{k=1}^N \lambda^{(N-k)} o_{i,k}}{\sum_{k=1}^N \lambda^{(N-k)}}, 0 < \lambda < 1,$$

where $o_{i,k} \in [0,1]$ represents the outcome score of the k -th observed event for host h_i , N is the number of observations included in the trust window, and λ is the temporal decay factor. Higher outcome scores represent more favorable host behavior, whereas lower scores represent events such as SLA violations, anomalies, or confirmed security incidents. The exponential weighting assigns greater importance to recent observations while progressively reducing the influence of older events. Although recency weighting improves the responsiveness of TS_i , the Trust Score remains dependent on historical observations. Therefore, DTD is maintained as a separate component to capture rapidly changing security conditions that may not yet be reflected in the host's historical trust profile.

The Vulnerability Assessment quantifies known host exposure using two normalized components: unpatched vulnerability exposure and unnecessary service or port exposure:

$$VA_i = \gamma \left(\frac{U_i}{P_i} \right) + (1 - \gamma) \left(\frac{E_i}{E_{\max}} \right), 0 \leq \gamma \leq 1,$$

where U_i denotes the number of known unpatched vulnerabilities associated with host h_i , P_i represents the total number of applicable vulnerability or patch items considered for that host, E_i is the number of unnecessarily exposed services or ports relative to the hardened configuration baseline, and $E_{\max}$ is the maximum exposure value used for normalization. The component ratios are bounded to $[0, 1]$, ensuring that

$$0 \leq VA_i \leq 1.$$

Higher values of VA_i therefore indicate greater vulnerability exposure.

Dynamic Threat Detection captures short-term evidence of potentially malicious or anomalous activity. It is defined as

$$DTD_i = \eta A_i + (1 - \eta) I_i, 0 \leq \eta \leq 1,$$

where $A_i \in [0,1]$ represents the normalized anomaly score derived from recent host or network telemetry and $I_i \in [0,1]$ represents the normalized threat-intelligence indicator score associated with host h_i . The parameter η controls the relative contribution of observed anomalous activity and threat-intelligence information. Consequently,

$$0 \leq DTD_i \leq 1,$$

with higher values indicating stronger evidence of an active or emerging security threat.

Together, TS_i , VA_i , and DTD_i provide complementary temporal perspectives on host security. TS_i represents

historical behavior, VA_i represents the current known vulnerability state, and DTD_i represents rapidly changing threat conditions. Their integration into SS_i therefore provides a normalized and interpretable measure of host security posture for use in subsequent allocation decisions.

8. Algorithm Design

SARAF employs a greedy placement strategy with bounded lookahead to balance security-aware host selection with resource distribution. For each incoming VM request, the algorithm first identifies all resource-feasible hosts and calculates their Allocation Scores. Rather than immediately selecting the host with the highest Allocation Score, SARAF retains the top K candidates and applies a secondary load-balancing criterion before making the final placement decision. This refinement reduces repeated placement on a small subset of highly ranked hosts while preserving the security-aware objective.

Algorithm 1: SARAF VM Placement

Input: VM request v_j with resource demand and sensitivity σ_j ; host set H ; Security Scores SS_i ; Resource Availability values RA_i ; parameters α , δ , ω , and lookahead width K .

Output: Selected host h^* or REJECT.

1. Candidates $\leftarrow \{h_i \text{ in } H : \text{Feasible}(h_i, v_j) = \text{TRUE}\}$
2. IF Candidates is empty THEN
3. RETURN REJECT
4. END IF
5. $\alpha_j \leftarrow \text{clamp}(\alpha + \sigma_j * \delta, 0, 1)$
6. FOR EACH h_i in Candidates DO
7. $AS_{ij} \leftarrow \alpha_j * SS_i + (1 - \alpha_j) * RA_i$
8. END FOR
9. TopK $\leftarrow K$ hosts in Candidates with the highest AS_{ij}
where $K \leftarrow \min(K, |\text{Candidates}|)$
10. FOR EACH h_i in TopK DO
11. $\text{Proj}_i \leftarrow \text{ProjectedUtilizationAfterPlacement}(h_i, v_j)$
12. $\text{Balance}_i \leftarrow 1 - |\text{Proj}_i - U_{\text{mean}}|$
13. $\text{FinalScore}_i \leftarrow \omega * AS_{ij} + (1 - \omega) * \text{Balance}_i$
14. END FOR
15. $h^* \leftarrow \text{argmax}_{(h_i \text{ in TopK})} \text{FinalScore}_i$
16. Place v_j on h^*
17. Update the resource state and RA_i of h^*
18. Record the placement outcome for subsequent Trust Score and Security Score updates
19. RETURN h^*

Here, α_j represents the effective security weight for VM v_j , determined by the base security weight α , workload sensitivity σ_j , and sensitivity adjustment parameter δ . The function $\text{clamp}(\cdot)$ restricts the resulting value to $[0, 1]$. Consequently, workloads with greater security

sensitivity can place greater emphasis on the Security Score during host ranking.

After the initial ranking, only the K hosts with the highest Allocation Scores are retained for secondary evaluation. For each candidate, $Proj_i$ represents the projected utilization of host h_i after placement of v_j , while U_{mean} represents the current mean utilization across the host pool. The balance term is defined as

$$Balance_i = 1 - |Proj_i - U_{\text{mean}}|.$$

Because utilization values are normalized to $[0, 1]$, $Balance_i$ also lies within $[0, 1]$, with higher values indicating that the projected host utilization remains closer to the current cluster-wide mean.

The final candidate score is calculated as

$FinalScore_i = \omega AS_{ij} + (1 - \omega) Balance_i$, $0 \leq \omega \leq 1$, where ω controls the relative importance of the security-resource Allocation Score and the secondary load-balancing criterion. A relatively high value of ω preserves the primary influence of AS_{ij} , while the balance component differentiates among the highest-ranked candidates according to their projected utilization.

For each incoming VM, feasibility checking and Allocation Score computation require $O(|H|)$ time. Selection of the top K candidates can be implemented in $O(|H| \log K)$ time, followed by $O(K)$ evaluation of the retained candidates. The resulting per-request complexity is therefore

$$O(|H| \log K),$$

which approaches linear behavior with respect to the number of hosts when K is fixed and small. This computational profile supports the use of SARAF for online placement while avoiding exhaustive optimization across all possible VM-to-host assignments.

Security-state updates operate alongside the placement procedure. Vulnerability information is refreshed periodically according to the configured assessment interval, while DTD is updated more frequently using recent telemetry. In addition, a significant security event triggers immediate recomputation of the affected host's Security Score. This event-driven mechanism enables subsequent allocation decisions to incorporate newly detected security conditions without waiting for the next scheduled vulnerability or trust update.

9. Methodology

The proposed framework was evaluated through a four-stage simulation methodology designed to assess its behavior under different workload conditions while maintaining consistent experimental settings across all scheduling approaches.

Stage 1: Environment Construction. A simulated cloud data center was configured with a heterogeneous pool of physical hosts. Each host was assigned an initial

synthetic security profile representing variations in trust, vulnerability exposure, and dynamic threat conditions. This configuration was designed to model differences in host security posture while providing a controlled environment for evaluating security-aware placement.

Stage 2: Workload Generation. Three workload scenarios, light, moderate, and bursty, were generated with different VM arrival rates, resource-demand distributions, and SLA-sensitivity distributions. These scenarios were used to evaluate scheduler behavior under varying levels of resource demand and contention.

Stage 3: Comparative Execution. Identical workload traces were executed using SARAF and each of the four baseline scheduling approaches. Maintaining the same workload traces across schedulers ensured that observed differences were attributable to the allocation strategies rather than variations in workload composition or arrival patterns.

Stage 4: Metric Collection and Analysis. For each scheduler-workload combination, the evaluation metrics defined in Section 11 were collected over the complete simulation run. The resulting measurements were aggregated and used for the comparative analyses presented in Sections 12 and 13.

Security-related outcomes were evaluated using controlled simulated compromise-event injection. A predefined subset of hosts was designated as compromised at randomized points during each simulation run, independently of the scheduling algorithm being evaluated. These events provided a common ground truth for assessing security exposure across schedulers. Host Placement Risk Rate (HPRR) was subsequently determined from the proportion of VM placements associated with hosts designated as compromised during the relevant evaluation period. Using the same compromise-event conditions across scheduling approaches enabled direct comparison of their ability to reduce VM exposure to compromised hosts within the simulated environment.

10. Experimental Setup

The proposed framework was implemented and evaluated using CloudSim [14]. The default broker and VM allocation mechanisms were extended to incorporate the SARAF security-scoring and placement components. The experimental environment was configured to evaluate SARAF under heterogeneous host capacities, different workload intensities, and controlled security events.

Data Center Configuration. The simulated data center consisted of 50 heterogeneous physical hosts divided into three capability tiers. Tier A consisted of 20 hosts, each configured with 16 vCPUs, 64 GB RAM, and 10 Gbps bandwidth. Tier B consisted of 20 hosts with 8 vCPUs, 32 GB RAM, and 5 Gbps bandwidth, while Tier C consisted of 10 hosts with 4 vCPUs, 16 GB RAM, and

1 Gbps bandwidth. This heterogeneous configuration was used to represent variation in computational capacity across the host pool.

Each host was assigned an initial synthetic security profile. The initial Trust Score TS_i was sampled from a truncated normal distribution with mean $\mu = 0.75$ and standard deviation $\sigma = 0.12$, with values restricted to $[0, 1]$. The initial Vulnerability Assessment VA_i was sampled from a uniform distribution over $[0.05, 0.45]$. The initial Dynamic Threat Detection value DTD_i was set to 0 for all hosts, representing the absence of active threat indicators at the beginning of each simulation.

Virtual Machine Configuration. Each simulation included 300 VM requests. VM arrivals followed a Poisson process, with arrival rates determined by the workload scenario [15]. VM resource demands were divided into three categories: small VMs requiring 1 vCPU and 2 GB RAM (45%), medium VMs requiring 2 vCPUs and 4 GB RAM (35%), and large VMs requiring 4 vCPUs and 8 GB RAM (20%). VM requests were also assigned SLA-sensitivity values: critical workloads with $\sigma_j = 0.9$ (15%), standard workloads with $\sigma_j = 0.5$ (55%), and best-effort workloads with $\sigma_j = 0.1$ (30%).

Workload Configuration. Three workload scenarios were evaluated. The light workload used a mean inter-arrival time of 12 simulated seconds and represented relatively low resource contention. The moderate workload used a mean inter-arrival time of 6 simulated seconds. The bursty workload used a base mean inter-arrival time of 6 simulated seconds, with four 90-second high-intensity periods during which the mean inter-arrival time was reduced to 1 simulated second. This configuration was designed to evaluate scheduler behavior under both steady and temporarily elevated demand.

SARAF Parameter Configuration. The bounded-lookahead procedure used $K = 5$ candidate hosts. The final candidate-ranking function assigned a weight of $\omega = 0.85$ to the Allocation Score and $1 - \omega = 0.15$ to the secondary load-balancing component. These parameter settings were maintained consistently across workload scenarios and comparative runs.

Compromise-Event Injection. Security events were introduced independently of the scheduling algorithm. Six hosts, corresponding to 12% of the host pool, were designated as compromised at randomized points during each simulation. Host selection for compromise was weighted according to the initial VA_i , such that hosts with greater vulnerability exposure had a higher probability of being selected. This weighting represents a simulation assumption rather than an empirically estimated compromise probability. The compromise state was not directly available to the scheduling algorithms. Instead, SARAF could respond only through

subsequent changes in its DTD and TS components, thereby preventing direct knowledge of future compromise events from influencing placement decisions.

Evaluation Process. Each scheduling algorithm was evaluated under each of the three workload scenarios using 10 randomized repetitions. Across repetitions, VM arrival timing, resource-demand sampling, and compromise-event timing were varied. The same experimental conditions were applied to the scheduling approaches within each repetition to support direct comparison. Results are reported as mean values across the 10 repetitions, with standard deviations reported to characterize between-run variability.

11. Performance Metrics

Six performance metrics were used to evaluate each scheduler-workload combination:

1. **High-Risk Placement Rate (HRPR):** The percentage of placed VMs assigned to a host with $SS_i < 0.5$ at the time of placement or residing on a host that was subsequently designated as compromised during the VM's residency. HRPR represents the primary security-oriented evaluation metric and measures the extent to which each scheduling strategy exposes workloads to hosts with elevated security risk.
2. **Resource Utilization Index (RUI):** The cluster-wide resource utilization efficiency calculated according to the formulation defined in Section 7.3. RUI captures the effectiveness with which the scheduling strategy utilizes available CPU, memory, and bandwidth resources.
3. **SLA Violation Rate:** The percentage of placed VMs that experienced a resource-contention event during their residency, resulting in degradation of the allocated computational resources or expected service performance.
4. **Placement Latency:** The average wall-clock time required to complete a VM placement decision. This metric measures the computational overhead introduced by each scheduling strategy.
5. **Load Distribution Variance:** The variance in host utilization across the physical host pool after placement. Lower variance indicates a more balanced distribution of workload across available hosts.
6. **Rejected Request Rate:** The percentage of incoming VM requests that could not be assigned to any resource-feasible host. This metric captures the effect of the scheduling strategy on admission capacity under different workload conditions.

These metrics jointly assess the security, resource efficiency, service quality, computational overhead, load distribution, and admission behavior of the evaluated scheduling approaches.

12. Results and Discussion

Table 1 summarizes the mean performance of the five scheduling approaches across 10 repetitions under the moderate workload, which was selected as the representative scenario for detailed comparison.

Scheduler	HRPR (%)	RUI	SLA Violation (%)	Placement Latency (ms)	Load Variance	Rejected (%)
FCFS	24.7	0.71	9.8	0.4	0.184	3.1
Round Robin	22.3	0.69	7.2	0.5	0.061	2.4
Traditional Load Balancing	19.1	0.76	5.4	1.1	0.048	1.9
Trust-Based Allocation	14.0	0.73	6.1	1.6	0.070	2.6
SARAF (proposed)	8.7	0.74	5.0	2.3	0.052	2.2

Under the moderate workload, SARAF achieved the lowest High-Risk Placement Rate (HRPR), reducing HRPR to 8.7% compared with 24.7% for FCFS, 22.3% for Round Robin, 19.1% for Traditional Load Balancing, and 14.0% for Trust-Based Allocation. Relative to Round Robin, this corresponds to an approximately 61% reduction in HRPR, while the reduction relative to Trust-Based Allocation was approximately 38%. These results indicate that incorporating vulnerability exposure and dynamic threat information in addition to historical trust can reduce high-risk VM placements within the simulated environment. The difference between SARAF and Trust-Based Allocation is particularly relevant to the proposed framework. Trust-based scheduling relies primarily on historical observations and may therefore respond gradually to newly emerging security events. SARAF supplements historical trust with VA and DTD components, allowing changes in vulnerability and threat conditions to influence subsequent placement decisions more rapidly. The observed HRPR reduction is consistent with the intended function of this multi-signal security assessment. Resource utilization showed a smaller difference among the stronger-performing schedulers. Traditional Load Balancing achieved the highest RUI of 0.76, followed by SARAF at 0.74 and Trust-Based Allocation at 0.73. Thus, SARAF maintained resource utilization close to the best-performing resource-oriented baseline while simultaneously achieving a substantially lower HRPR. FCFS and Round Robin produced lower RUI values of 0.71 and 0.69, respectively. These findings indicate that the security-aware placement objective introduced a modest resource-utilization trade-off relative to Traditional Load Balancing.

SARAF also achieved the lowest SLA violation rate at 5.0%, compared with 5.4% for Traditional Load Balancing, 6.1% for Trust-Based Allocation, 7.2% for Round Robin, and 9.8% for FCFS. The relatively low SLA violation rate suggests that incorporating security information into host selection did not result in substantial degradation of service-related performance under the moderate workload. The additional

computations required by SARAF were most apparent in placement latency. SARAF required an average of 2.3 ms per placement decision, compared with 0.4 ms for FCFS, 0.5 ms for Round Robin, 1.1 ms for Traditional Load Balancing, and 1.6 ms for Trust-Based Allocation. Although SARAF therefore introduced greater computational overhead than the baseline methods, the measured latency remained in the millisecond range under the simulated configuration. This overhead is attributable to the additional security-score processing, candidate ranking, and bounded-lookahead evaluation performed during placement.

Load Distribution Variance for SARAF was 0.052, close to the 0.048 obtained by Traditional Load Balancing and lower than the values observed for FCFS (0.184), Round Robin (0.061), and Trust-Based Allocation (0.070). This result indicates that the secondary balance component of the SARAF placement algorithm maintained a relatively even distribution of workloads despite the additional emphasis on host security. The rejected request rate under the moderate workload was 2.2% for SARAF, compared with 1.9% for Traditional Load Balancing, 2.4% for Round Robin, 2.6% for Trust-Based Allocation, and 3.1% for FCFS. Thus, the security-aware placement strategy did not produce a substantial increase in rejected requests under moderate workload conditions.

Under the light workload, differences in resource-oriented metrics were reduced because lower contention provided greater placement flexibility. However, SARAF continued to exhibit a lower HRPR than the baseline approaches, indicating that its security-oriented host ranking remained active even when resource contention was limited.

Under the bursty workload, SARAF's rejected request rate increased to 4.6% during high-intensity burst periods, compared with 2.2% under the overall moderate workload. Trust-Based Allocation reached a burst-window rejection rate of 5.1%. The increased rejection rate under burst conditions indicates that reduced resource availability and the cumulative effects of earlier placement decisions limit admission capacity as contention increases. This behavior highlights the

importance of evaluating security-aware scheduling not only in terms of security exposure but also in terms of resource availability and admission performance under high-demand conditions.

Overall, the results demonstrate that SARAF achieved the strongest security-oriented performance among the evaluated approaches while maintaining resource utilization, SLA performance, load distribution, and request acceptance at levels comparable to the stronger baseline schedulers. The principal trade-off was increased placement latency resulting from the additional security-scoring and bounded-lookahead operations. These findings support the use of integrated security signals in VM placement while also emphasizing the need to consider computational overhead and resource contention when deploying security-aware scheduling strategies.

13. Comparative Analysis with Existing Approaches

Comparison of SARAF with the four baseline scheduling approaches demonstrates distinct trade-offs among security awareness, resource efficiency, and computational overhead. FCFS and Round Robin provide low-complexity placement mechanisms but do not explicitly incorporate host security information into their allocation decisions. Consequently, their comparatively higher HRPR values reflect the absence of security-oriented host assessment during VM placement. Traditional Load Balancing improves resource utilization, load distribution, and SLA-related performance but similarly does not explicitly consider host security posture.

Trust-Based Allocation provides a more relevant security-oriented baseline because it incorporates historical host behavior into placement decisions. Under the moderate workload, Trust-Based Allocation achieved an HRPR of 14.0%, compared with 8.7% for SARAF. The lower HRPR observed with SARAF is consistent with the additional incorporation of vulnerability exposure and dynamic threat information alongside historical trust. Within the simulated environment, these additional security signals enabled SARAF to respond to host security conditions that may not have been fully represented by historical trust alone.

The comparative results also demonstrate that SARAF does not dominate every baseline across all evaluation dimensions. Traditional Load Balancing achieved the highest RUI (0.76) and the lowest Load Distribution Variance (0.048), compared with 0.74 and 0.052, respectively, for SARAF. SARAF also exhibited greater placement latency than the baseline approaches because of the additional security-score computation, candidate ranking, and bounded-lookahead evaluation. These results demonstrate a measurable trade-off between enhanced security-aware placement and computational or resource-management efficiency.

Trust-Based Allocation may represent a less computationally demanding security-aware alternative

when continuously updated vulnerability and threat information is unavailable. In contrast, SARAF depends on the availability and timely updating of the security signals used to calculate VA and DTD. Therefore, the benefits of the proposed framework depend partly on the quality, availability, and freshness of the underlying security information.

The weighting structure of SARAF also permits the contribution of individual security components to be adjusted according to data availability. For example, setting

$$w_v = w_d = 0$$

removes the vulnerability and dynamic threat components from the Security Score, leaving historical trust as the security-related input. In such a configuration, however, the additional benefits associated with vulnerability and dynamic threat information would no longer be available. Thus, the primary contribution of SARAF lies in its ability to integrate multiple complementary security signals within a unified and workload-sensitive allocation framework rather than relying exclusively on a single security indicator.

Overall, the comparative analysis indicates that SARAF is most advantageous when security-sensitive workloads require placement decisions that account for both historical and current host security conditions. Its improved security-oriented performance is accompanied by modest resource-efficiency and computational trade-offs, emphasizing that the selection of an allocation strategy should reflect the security requirements, resource constraints, and monitoring capabilities of the target cloud environment.

14. Limitations

Several limitations of the present study should be considered when interpreting the findings.

First, the compromise-event injection model used in the simulation represents a simplified approximation of real-world security incidents. In operational cloud environments, host compromise is not necessarily immediately observable or represented by an explicit event indicator. Furthermore, the probability of simulated compromise was weighted according to the initial Vulnerability Assessment (VA_i). Although this design provides a controlled mechanism for evaluating security-aware placement, the relationship between vulnerability exposure and compromise probability was a modeling assumption and was not estimated from empirical incident data.

Second, the Dynamic Threat Detection (DTD) component assumes access to sufficiently timely host-level security telemetry, including authentication activity, network-connection information, anomaly indicators, and threat-intelligence data. The availability, granularity, and update frequency of such information may vary across cloud environments. Delayed, incomplete, or unavailable telemetry could reduce the

responsiveness of DTD and consequently diminish the advantage of SARAF over approaches based primarily on historical trust.

Third, the placement-latency overhead observed in the simulated environment has not been evaluated at production-scale data-center sizes. Although the proposed algorithm exhibits near-linear behavior for a fixed and small lookahead width K , substantially larger host pools may increase candidate-evaluation overhead. Additional mechanisms such as host pre-filtering, hierarchical scheduling, or candidate pruning may therefore be required for deployment in large-scale cloud environments.

Fourth, the weighting parameters used by SARAF were selected through manual sensitivity analysis rather than formal workload-specific optimization. Consequently, the parameter configuration used in this study should not be interpreted as globally optimal. Alternative parameter settings may alter the balance among security exposure, resource utilization, placement latency, load distribution, and other performance outcomes. More systematic parameter optimization and sensitivity analysis are therefore required to determine appropriate configurations for different operational environments and workload classes.

Fifth, the current framework makes placement decisions immediately and does not incorporate a queuing or deferred-retry mechanism for requests that cannot be accommodated at the time of arrival. Under bursty workload conditions, reduced resource availability and the cumulative effects of previous placement decisions increased the rejected request rate. Incorporating workload-aware queuing, deferred placement, or retry mechanisms could improve admission performance, particularly for non-critical workloads that can tolerate additional placement delay.

Finally, the effectiveness of SARAF depends on the quality and timeliness of the security information used to calculate its component scores. When vulnerability information or dynamic threat telemetry is sparse, unreliable, or substantially delayed, the framework may provide less benefit while retaining additional computational overhead. Therefore, the applicability of SARAF should be evaluated in relation to the monitoring capabilities and security-data availability of the target cloud environment.

15. Conclusion

This study proposed the Security-Aware Resource Allocation Framework (SARAF), which integrates historical trust, vulnerability exposure, and dynamic threat information into VM placement decisions. Unlike resource-oriented scheduling approaches that primarily consider computational capacity, SARAF combines these complementary security signals into a composite Security Score and integrates this score with Resource Availability through a workload-sensitive Allocation Score. CloudSim-based evaluation against FCFS, Round

Robin, Traditional Load Balancing, and Trust-Based Allocation demonstrated that SARAF achieved the lowest High-Risk Placement Rate among the evaluated approaches. Under the moderate workload, SARAF achieved an HRPR of 8.7%, representing an approximately 38% relative reduction compared with Trust-Based Allocation, while maintaining an RUI of 0.74 compared with 0.76 for Traditional Load Balancing. SARAF also maintained competitive SLA violation, load-distribution, and rejected-request rates, although these improvements were accompanied by increased placement latency associated with additional security-score computation and bounded-lookahead evaluation. Overall, the findings demonstrate the potential value of incorporating security information with different temporal characteristics into resource-allocation decisions. Historical trust provides information about previous host behavior, vulnerability assessment represents current known exposure, and dynamic threat detection captures more rapidly changing security conditions. Integrating these signals enables placement decisions to account for host security posture alongside resource availability rather than treating security as an independent downstream process.

The present findings are based on a controlled simulation environment and should therefore be interpreted within the limitations of the experimental design. Future work should focus on adaptive parameter optimization, improved admission handling under burst conditions, scalability to larger cloud environments, and validation using real-world security and operational data. These developments could further clarify the practical applicability of security-aware resource allocation in dynamic multi-tenant cloud environments.

16. Future Scope

The limitations identified in this study provide several directions for future development of SARAF. First, adaptive or learning-based parameter optimization could replace manually selected weights with values that respond dynamically to workload and security conditions. Reinforcement learning or related approaches could be investigated to optimize the Security Score weights (w_t, w_v, w_d) and Allocation Score parameters (α, δ) using an objective that jointly considers security exposure and resource utilization.

Second, a deferred-placement mechanism could be introduced for best-effort and low-sensitivity workloads. Rather than immediately rejecting requests when sufficient resources are unavailable, such requests could be temporarily queued and reconsidered as host capacity becomes available. This extension may improve admission performance under bursty workload conditions while preserving stronger security preferences for critical workloads.

Third, the scalability of SARAF could be improved through hierarchical candidate selection or host pre-filtering. Hosts could initially be filtered according to

resource feasibility and coarse security characteristics before the complete scoring and bounded-lookahead procedure is applied. Such mechanisms may reduce placement overhead when SARAF is extended to substantially larger host pools.

Finally, future studies should validate the proposed security model using real-world or anonymized cloud-security data where such information is available under appropriate data-sharing and privacy requirements. In particular, empirical incident data could be used to evaluate the assumptions underlying compromise-event injection, refine the relationship between vulnerability exposure and compromise probability, and assess the practical validity of HRPR as a security-oriented placement metric. Evaluation in larger-scale and operational cloud environments would further establish the generalizability of the proposed framework.

References

1. Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, Zaharia M. A view of cloud computing. *Communications of the ACM*. 2010 Apr 1;53(4):50-8.
2. Kumar M, Sharma SC, Goel A, Singh SP. A comprehensive survey for scheduling techniques in cloud computing. *Journal of Network and Computer Applications*. 2019 Oct 1;143:1-33.
3. Agarwal A, Duong TN. Secure virtual machine placement in cloud data centers. *Future Generation Computer Systems*. 2019 Nov 1;100:210-22.
4. Huang J, Nicol DM. Trust mechanisms for cloud computing. *Journal of Cloud Computing: Advances, Systems and Applications*. 2013 Apr 24;2(1):9.
5. Bittencourt LF, Goldman A, Madeira ER, da Fonseca NL, Sakellariou R. Scheduling in distributed systems: A cloud computing perspective. *Computer science review*. 2018 Nov 1;30:31-54.
6. Tian WH, Xu MX, Zhou GY, Wu K, Xu CZ, Buyya R. Prepartition: load balancing approach for virtual machine reservations in a cloud data center. *Journal of Computer Science and Technology*. 2023 Jul;38(4):773-92.
7. Shafiq DA, Jhanjhi NZ, Abdullah A. Load balancing techniques in cloud computing environment: A review. *Journal of King Saud University Computer and Information Sciences*. 2022 Jul;34(7):3910-33.
8. Rjoub G, Bentahar J, Wahab OA. BigTrustScheduling: Trust-aware big data task scheduling approach in cloud computing environments. *Future Generation Computer Systems*. 2020 Sep 1;110:1079-97.
9. Marudhadevi D, Dhatchayani VN, Sriram VS. A trust evaluation model for cloud computing using service level agreement. *The Computer Journal*. 2015 Oct 1;58(10):2225-32.
10. Han J, Zang W, Liu L, Chen S, Yu M. Risk-aware multi-objective optimized virtual machine placement in the cloud. *Journal of Computer Security*. 2018 Aug 9;26(5):707-30.
11. Shetty S, Yuchi X, Song M. Security-aware virtual machine placement in cloud data center. In *Moving Target Defense for Distributed Systems 2016* Apr 21 (pp. 13-24). Cham: Springer International Publishing.
12. Zhou G, Tian W, Buyya R, Xue R, Song L. Deep reinforcement learning-based methods for resource scheduling in cloud computing: a review and future directions. *Artificial Intelligence Review*. 2024 Apr 23;57(5):124.
13. Hou H, Jawaddi SN, Ismail A. Energy efficient task scheduling based on deep reinforcement learning in cloud environment: A specialized review. *Future Generation Computer Systems*. 2024 Feb 1;151:214-31.
14. Calheiros RN, Ranjan R, Beloglazov A, De Rose CA, Buyya R. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and experience*. 2011 Jan;41(1):23-50.
15. Vakiliinia S, Qiu D, Ali MM. Optimal multi-dimensional dynamic resource allocation in mobile cloud computing. *EURASIP Journal on Wireless Communications and Networking*. 2014 Nov 28;2014(1):201.