Enhancing Security in Cloud-Based VM Migration: A Trust-Centric Hybrid Optimization Approach

Gokul Narayanan  
Department of Information Security, School of Computer Science and Engineering, VIT University, India  
gokulgn@gmail.com

Saravanan Guru Kannan  
Department of Information Security, School of Computer Science and Engineering, VIT University, India  
saravankan@gmail.ac.in

Abstract: Virtual Machine (VM) migration in cloud computing is essential to accomplishing various resource management goals, such as load balancing, power management, and resource sharing. Ensuring Quality of Service (QoS) is crucial while migrating VMs, which means that VMs must run continuously during the procedure. The dynamic nature of resource requirements in cloud computing, driven by “service-on-demand,” poses security risks, even while live VM migration mechanisms help to maintain VM availability. This paper addresses changing resource requirements and improves overall system security by introducing a novel method for safe live VM migration. In this paper, a robust optimization approach has been proposed for live VM migration named the Gorilla-based Shuffled Shepherd Optimization Approach (GBSSOA). The proposed approach maximizes the number of fitness targets, such as migration time, QoS parameters, and job runtime. The method takes into account trust values in addition to the previously described performance measures, with a focus on security. Key performance indicators QoS as workload, migration time, trust, and GBSSOA-based secure live VM migration are taken into account during the evaluation. The obtained values of 0.141, 0.654, 342.254ms, and 0.569, respectively, show noteworthy accomplishments in the results. These results highlight how well the strategy developed may simultaneously improve performance metrics and strengthen the security of live VM migration in dynamic cloud computing environments.

Keywords: Virtual machine, migration, downtime, migration time, security, optimization algorithm, gorilla-based shuffled shepherd optimization approach.

Received June 12, 2023; accepted December 20, 2023
https://doi.org/10.34028/iajit/21/1/15

1. Introduction

In the present technological era, cloud computing has reshaped the way users access data and other resources. Cloud computing is an emerging technology for providing services to clients irrespective of the location or time. Recently, cloud computing has found applications in many aspects of our day-to-day life including markets, business, industry, enterprise, and government. The services in the cloud are handled by a third party called a Cloud Service Provider (CSP) [22]. Cloud computing is a type of distributed parallel system that allows access to resources, processing tasks, data, centralized data storage, and servers that are available on remote servers [20]. The CSP offers services to the clients on a lease basis rather than buying it, thereby achieving reduced software cost [2]. Moreover, cloud computing utilizes remote computers for storage and retrieval of data rather than storing it in local computers [32]. Cloud computing normally utilizes public clouds which allow consolidation of numerous resources within a few PMs. The CSPs perform the process of handling the resources by scaling them based on the dynamic varying loads corresponding to the consumer needs [8]. Virtualization is the key aspect that allows multiple users to share the same resources by providing virtual computing resources and storage facility to the users. Virtualization helps in creating numerous instances of the Virtual Machine (VM) on a Physical Machine (PM) thereby enhancing the utilization of resources and enhancing the profits of the CSP [4]. Virtualization is done with the aim of managing the workload by transforming the conventional computing systems to make them economical, efficient, and scalable [15].

Virtualization facilitates the running of numerous Operating Systems (OS) on a single PM. A hypervisor controls every OS that runs on the VM. A key advantage of virtualization is the efficient management of the resources, which is accomplished by migration. The VMs at one PM (source) can be easily migrated to another PM (destination), and at the same time, the VM is kept operational throughout the process. By performing migration of VMs to the underloaded PMs from the overloaded ones, load balancing can be achieved [17]. The VMs constantly require extra resources during migration which impacts the efficiency of the currently running applications. thus, the migration process should be finished within a short span with the help of optimal bandwidth and a targeted server, thus improving the transparency and performance of the process [6]. VM migration can be carried out using two approaches, such as offline as well as live migration. If there is a delay or absence of active connections during
the migration, then the user may get disconnected from the services provided by the CSP. Thus, in order to avoid this kind of scenario, live migration is employed [18]. The VMs are migrated between the PMs by using dedicated or shared resources by the hypervisor with the help of live migration. As the VMs are migrated without interrupting the process during live migration, optimal utilization of resources is achieved. Further, the process ensures uninterrupted connectivity as well as helps to meet the Service Level Agreement (SLA) [6]. Live migration ensures that the CSP can provide a profitable way to access CPU and the other resources [21]. The study focuses on the methods and developments that enable smooth VM migrations in a live environment, resulting in better system performance and effective resource use [3].

Security is a key issue that must be addressed while the migration is carried out. In the case of a Wide Area Network (WAN), the transfer of VMs happens over heterogeneous links, which are shared thereby increasing the vulnerability of VMs. As the VMs contain sensitive information, like passwords, encryption keys, and so on, they must be secluded and protected during the process of migration [28]. Another major issue affecting the performance of the cloud system is resource allocation. The cloud must provide the customers with high quality resources in all scenarios for achieving maximum utility. Resource allocation is performed by mapping the VMs to the PMs and is done by using two processes, such as mapping and scheduling. Resource allocation offers the advantage of load balancing, minimal request execution time, and efficient bandwidth utilization. Normally, efficient VM resource provisioning ensures profit of CSP is enhanced by delivering the services corresponding to the varying user demands and minimizes the cost of CSP by reducing the energy consumed by the PMs [26]. Though virtualization effectively minimizes the hardware cost, the energy consumed during the process is high. Moreover, when the VM is overloaded, it won’t be able to execute all services. Thus, efficient ways of allocating resources have to be considered for reducing the energy consumed [29]. Further, performing live VM migration with maximum Quality of Service (QoS) and minimal time is highly challenging, and the QoS and migration time depends on the bandwidth allocated. Several live VM migration algorithms have been proposed, such as Fast Transparent Migration (FTM) [16], Remote Direct Memory Access (RDMA) [9], multi-objective optimized replica placement approach [13], model-driven scheme [14], and so on.

This paper presents a Gorilla-based Shuffled Shepherd Optimization Approach (GBSSOA)-based secure live VM migration technique for providing security during live VM migration. Live VM migration is accomplished considering the workload of the PM computer depending on the QoS and migration time. Once the workload is computed, the value is compared with the threshold to determine if VM migration is required or not. If required, VM migration is performed with the presented GBSSOA, considering objectives, like migration time, run time of the task, and QoS. Further, the trust value is used to ensure security during the VM migration.

The key contribution of this study is listed below.

- Introduced GBSSOA for Live VM Migration: a new optimization algorithm named GBSSOA is devised for performing VM migration, wherein the GBSSOA is developed by adapting the exploration capability of the Gorilla Troops Optimization (GTO) in accordance with the Shuffled Shepherd Optimization Algorithm (SSOA) for improving the optimization process, based on objectives, such as run time of the task, migration time and QoS.

The rest of the study has the following organizational structure; the upcoming section details the related work in live VM migration; section 3 explains on the system model of the cloud, section 4 details the devised live VM migration scheme, section 5 elucidates the results obtained and the evaluation of the devised technique, and the research work is concluded in section 6.

2. Motivation

There has been a growing focus on cloud computing recently, with one of the major focuses on the service migration between the clouds. VM migration ensures the availability and reliability of the service provided by cloud computing. Though various approaches have been developed for achieving VM migration, the security issues associated with the live VM migration have been not much researched. This section details the prevailing schemes of VM migration together with their qualities and challenges, which enthused the formation of the devised scheme.

2.1. Literature Review

Though several studies have addressed the issues of VM migration, we consider eight stat-of-art approaches in the analysis. Hu et al. [10] HMDC technique leverages hybrid memory copy and delta compression to enhance live VM migration performance. While demonstrably effective, limitations like increased CPU overhead and potential scalability challenges warrant further investigation for optimal application. Sun et al. [23] introduced Security Protection of Live Migration (SPLM) for providing security during migration using encryption and security policy transfer. The SPLM model was comprised of various modules, like Hypervisor Access Engine (HAE), VM Security Agent (SA), Virtual Security Gateway (VSG), and Centralized Management Platform (CMP), wherein the SA and CMP were responsible for providing security. The SPLM system was highly effective in handling huge
traffic while providing security during live migration, although the system suffered from high complexity and cost as the VM count increases. To minimize cost, Addya et al. [1], devised a Policy aware VM migration strategy for providing security during live VM migration in cloud federation. The cost associated during migration (down time and real migration time) and communication (request, reply, path establishment) was analyzed for parallel, serial, and enhanced serial techniques. Moreover, a technique for estimating the power consumed during the migration process was also developed. This scheme was effectively utilized for scrutinizing and identification of any abnormal incidents during the VM migration process, but it did not consider migration from one service provider to another. To overcome the drawback listed in [23], Zeb et al. [31], devised a Relative security metric model for securing VM migration. This framework utilized three quantitative security measures, like Cost/Benefit Measure (CBM), Performance Improvement Factor (PIF), and Attack Resiliency Measure (ARM) for estimating the effectiveness of the scheme. This enhanced security control scheme can effectively handle all attacks launched by Dolev-Yao (DY) model. However, it was vulnerable to some attacks, such as the ones produced by Key Compromise Impersonation (KCI) models. Attacks were effectively handled, where Torquato et al. [24] developed a Stochastic Reward Net (SRN) scheme for evaluating the security risk during the VM migration scheduling. The SRN model was implemented using three major modules, namely main node, VM and the standby node. This model effectively analyzed the tradeoff between security risk and availability during migration for Virtual Machine Monitor (VMM) rejuvenation by utilizing a security factor called Risk Score but was not suitable for multiple VMs.

Multiple simultaneous VM migration was considered, where Yang et al. [30] presented a Particle Swarm Optimization and Genetic Algorithm (PSO+GA) for allocating resources during VM migration. This method considered the constrained bandwidth allocation problem, while enhancing the QoS and satisfying the migration time. This scheme was highly effective in attaining an improved QoS with minimal migration time, although it failed to ensure enhanced performance as deadlines increased. This issue was overcome, where Deylami et al. [7] introduced a system named Kororā for providing security to the live VM migration process. Here, a virtual trusted platform was utilized for enhancing VM migration integrity and, VM migration was carried out using agents, like data organization, integrity analyzer, data plane, input/output, and Virtual Trust Platform Model (VTPM) agents. This scheme was efficient in providing trusted computing for migrating numerous VMs on the same platform, but it failed to improve the live VM migration integrity. The integrity was enhanced, where Huang et al. [11] proposed a Policy-Customized Trusted Cloud Service (PC-TCS) model for ascertaining the uniformity of customers’ policies during VM migration. The PC-TCS scheme was implemented using two modules, namely the blockchain-based VM-migration and Attribute-Based Signature (ABS)-based remote-attestation approach. This approach was successful in performing validation of the VM migration in satisfying the security policy and also in reviewing the security of the request, although the approach faced limitations when applying it to the real-time clouds. This issue was overcome, where in Verma [25] devised a Dual Conditional Moth Flame Algorithm (DC-MFA) for performing load balancing in cloud environment. Here, the load balancing was achieved by minimizing the load of the servers by considering selection of one or more VMs for migrating, based on objectives, such as resource cost, migration cost, make span, security, CPU utilization, and migration cost. The DC-MFA was successful in minimizing the resource load and had a fast convergence but was unsuccessful in minimizing the energy consumption. In order to solve the current cloud storage security methods, Bhat et al. [5] offered the Probabilistic Public Key Encryption Switching (PPKES) protocol, which offers a single platform for both homomorphic and non-malleable cloud applications. For intermediate ciphertext connections, it presents a unique discrete chain bit pair function and contiguous chain bit pair encryption. The comparison various VM migration models are provided in Table 1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Approach</th>
<th>Limitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security considerations</td>
<td>HMDC [10]</td>
<td>Lack of security</td>
</tr>
<tr>
<td>QoS and Load Balancing</td>
<td>SPLM Method [23]</td>
<td>Lack of consideration for QoS</td>
</tr>
<tr>
<td>Security Risk Evaluation</td>
<td>SRN Model [24]</td>
<td>Failure to consider various attacks</td>
</tr>
<tr>
<td>Integration of Agents</td>
<td>Kororā Scheme [7]</td>
<td>Inability to include agents like Libvirt and Go</td>
</tr>
<tr>
<td>Comprehensive Security Aspects</td>
<td>Prevailing Methodologies</td>
<td>Lack of consideration for security aspects</td>
</tr>
</tbody>
</table>

3. System Model

Cloud computing provides clients a means of accessing the shared pool of resources and computing devices regardless of their position and time. Virtualization is the key aspect of cloud, which enables in enhancing the availability of resources to the clients. Virtualization enables the cloud to create numerous instances of VM in single PM, wherein the VMs can be employed for processing distinct tasks. With the increase in user demands, the VMs has to be allocated with utmost care as inaccurate assignment of tasks can lead to poor performance as a result of underloading or overloading. The system model of the cloud is depicted in Figure 1.
Consider a cloud system comprising of $z$ PMs, and the set of PMs is expressed as $PM = \{A_1, A_2, \ldots, A_y, \ldots A_z\}$, where $A_y$ signifies the $y$th PM. Each PM is composed of a distinct number of VMs. Now, consider there are $x$ number of VMs in the $y$th PM and it can be represented as, $VM = \{B_1^y, B_2^y, \ldots, B_x^y\}$. Each VM is composed of a set of parameters given by $\{CPU_b, M_b, BW_b, MT_b, QoS_b\}$, wherein $CPU_b$ indicates the CPU of the $b$th VM, $M_b$ denotes the memory of the $b$th VM, $BW_b$ is the bandwidth of the $b$th VM, $MT_b$ specifies the migration time of the $b$th VM, and $QoS_b$ is the QoS of the $b$th VM.

4. Devised GBSSOA based Live VM Migration Scheme

This section elaborates on the developed secure live VM migration in cloud depending on resource allocation. The introduced GBSSOA based live migration scheme is implemented using the following steps as in Figure 2.

4.1. Algorithmic Steps

The algorithmic steps of the live VM migration scheme are listed in Algorithm (1) below:

**Algorithm 1: GBSSOA TaskAssignment**

**Input:** List of VMs, List of Tasks

**Output:** Task assignments to VMs using live VM migration if needed

**Procedure:**
1. Assign tasks to VMs in a round-robin manner initially.
2. Initialize iteration counter to 0.
3. Repeat until all tasks are assigned:
   - Calculate the workload $W_m$ of the Physical Machine (PM).
   - If $W_m > 0.9$ (threshold), then perform live VM migration using GBSSOA scheme to the underloaded PM.
   - Increment the iteration counter.
4. End procedure.

4.2. Prerequisites

The workload of the PM is computed based on VM parameters, such as migration time, and QoS of the VM, wherein the migration time depends on the memory, bandwidth, and CPU. The VM parameters, namely the QoS and migration time are detailed as follows:

- **Migration Time**: migration time of the VM denotes the time taken for performing migration of a VM from one cloud to another. The VM migration time is computed with the help of the pre-copy scheme and is represented as,

$$MT_P = \frac{M}{BW - D} \quad (1)$$

where $P$ indicates the index of the live VM migration task, $BW$ indicates the bandwidth of the network existing among the source as well as the destination hosts, $M$ indicates the memory of the VM, and the dirty rate is specified as $D$. Dirty rate indicates the data rate of the VM while migrating and is dependent on the currently running application’s QoS requirements.

- **QoS**: QoS is used to indicate the quality needed by the application running on the VM during migration. If an application needs higher QoS, then the VM workload estimated is then compared to the threshold, and if estimated workload is higher than the threshold, live VM migration is performed using the introduced GBSSOA. VM migration is carried out by moving the VM from the overloaded PM to the underloaded PM, based on objectives, like run time of the task, migration time and QoS. Here, the introduced GBSSOA is developed by adapting the GTO [27] in line with the SSOA [12]. Further, the security of the VM migration is ensured by considering the fitness considering the trust parameters of the VM, along with the objectives mentioned above.
memory has to be restructured with high data rate while migrating. The data rate is also known as service rate or dirty rate. QoS depends on the resource allocation, which is given by,
\[
RA = \max_{r_{p,q},D_{p,q},S_p} \frac{1}{P} \sum_{p=1}^{P} \left( 1 - \frac{1}{P} \sum_{p=1}^{P} C_{p,q} \right)
\]  
(2)
\[
C_{p,q} = f(D_{p,q}) \quad p = 1, 2, ..., P; q = 1, 2, ..., P
\]  
(3)
\[
\forall p \in [1, P]; \forall q \in [1, S_p], C_{p,q} \geq w
\]  
(4)
\[
\forall p \in [1, P]; \forall q \in [1, S_p], C_{p,q} \geq w
\]  
(5)
\[
\forall p \in [1, P]; \sum_{q=1}^{S_p} \left( D_{p,q} - D_{p,q} \right) = F_p
\]  
(6)
\[
\forall p \in [1, P]; S_p \geq f_p
\]  
(7)

Here, \(r_{p,q}\) and \(D_{p,q}\) denotes the bandwidth allocated and dirty rate for the task \(P\) in time slot \(q\), \(S_p\) indicates the actual migration time for task \(p\), \(P\) signifies the live VM migration task count in the total time slots, \(I\) represents the length of the time slot, \(C_{p,q}\) is the QoS of the task \(p\) in time slot \(q\), \(w\) denotes the minimal QoS required for the VM migration task, \(E_q\) is the bandwidth of the network in the time slot \(q\). \(E_q\) specifies the data transmission workload of the task \(p\) and is normally the VM memory size.

4.3. Workload Migration Using the Devised GBSSOA

Once the workload of the PM is computed, comparison of the workload is carried out with the threshold. If the computed workload of the PM is higher than the threshold then live VM migration is performed by migrating the VM from the overloaded PM to an underloaded PM for balancing the load.

Here, \(t\) signifies the number of VMs which needs to be migrated. Solution encoding refers to the pictorial representation of the solution of the devised scheme. The VM in the overloaded PM is migrated to the underloaded PM for balancing the load.

Here, \(t\) signifies the number of VMs which needs to be migrated. Solution encoding refers to the pictorial representation of the solution of the devised scheme. The VM in the overloaded PM is migrated to the underloaded PM for balancing the load.

Here, \(t\) signifies the number of VMs which needs to be migrated. Solution encoding refers to the pictorial representation of the solution of the devised scheme. The VM in the overloaded PM is migrated to the underloaded PM for balancing the load.

Here, \(t\) signifies the number of VMs which needs to be migrated. Solution encoding refers to the pictorial representation of the solution of the devised scheme. The VM in the overloaded PM is migrated to the underloaded PM for balancing the load.

Here, \(t\) signifies the number of VMs which needs to be migrated. Solution encoding refers to the pictorial representation of the solution of the devised scheme. The VM in the overloaded PM is migrated to the underloaded PM for balancing the load.
Here, $RT_{ij}$ indicates the run time of the task running in the $i^{th}$ VM under the $j^{th}$ PM, $MT_{ij}$, QoS$_{ij}$ and Trust$_{ij}$ denotes the migration time, QoS and trust parameter of the $i^{th}$ VM under the $j^{th}$ PM.

4.3.1. Devised GBSSOA

Live VM migration is carried out by using the developed GBSSOA, which is created by modifying the GTO [27] in line with the SSOA [12] for enhancing the performance of the optimization. The GTO algorithm is a gradient free optimization approach which is based on the way of life of gorillas. The gorillas live together as a group called troop that comprises of a silverback gorilla or a male along with multiple females with their offspring. The grownup male gorillas are referred to as silverbacks, and the smaller males are termed blackbacks. In GTO, exploration and exploitation are performed by five operators. In exploration phase, the gorillas perform actions, like moving to an unfamiliar area, moving to a familiar region, and migrating towards other gorillas, whereas in exploitation they tag along the silverback and compete for the female gorillas. The GTO is highly efficient in attaining improved convergence and is effective in solving multi-objective optimization issues, although the stability of GTO reduces with highly complex optimization issues. The SSOA, on the other hand, is based on the inspiration of the intuition of a shepherd. Here, the sheep are divided into various groups called herds and, in every herd, a sheep is chosen as shepherd and the ones with the best objective is chosen as horses. The shepherd attempts to guide the sheep to the horse, and the sheep and horses are moved for attaining the new location of the shepherd and based on the shepherd’s nature in the herd optimization is carried out. The SSOA is highly effective in attaining the best solution with minimal iterations and has high accuracy. However, the efficacy of the SSOA is highly oriented on the parameters. By combining the SSOA and GTO, the devised GBSSOA effectively overcomes the drawback of both and enhances the performance of optimization. The devised GBSSOA is implemented using the following steps.

1. Initialization: consider there are $C$ number of gorillas in a search space of dimension $K$, then the population of gorillas can be initialized as,

$$LC \times K = rd(C, K) \times (up - low) + low$$  \hspace{1cm} (16)

Here, $rd$ is a matrix with dimension $C \times K$ with elements having arbitrary values in the range [0,1], $up$ and $low$ designates the maximum and minimum bound of the search space.

2. Fitness evaluation: the optimal function is determined considering the fitness measure which is estimated by using Equation (13).

3. Exploration: generally, gorillas live in groups which are dominated by the silverback, which is responsible for the troop’s movement and safety, mediating fights decision making, and leading the members to the food sources. In some scenarios, the gorillas tend to leave their group, where in such cases they may migrate into known or unknown territory, or they may shift to other groups. The male gorillas may create a new troop by drawing the female gorillas, which have moved from their groups in certain cases. All gorillas are assumed to be the candidate solution and based on the optimization the silverback is found by determining the best solution. The exploration can be modelled using the following equation considering the three approaches discussed above.

$$QL_{ij} + 1 = \begin{cases} 
(m_1 - low) \times m_2 + low & m_1 < \lambda \\
(m_1 - \bar{K}) \times T(t) + \delta \times T(t) + \lambda & m_1 \geq 0.5 \\
L(t) - \delta \times S \times (L(t) - L_y(t)) + m_2 \times (L(t) - L_y(t)) & m_1 < 0.5
\end{cases}$$ \hspace{1cm} (17)

where, $\lambda$ is a constant that denotes the parameter for choosing the approach for migration to unfamiliar territory, $i$ indicates the present iteration, $Q(t+1)$ denotes the candidate location of the search agents iteration $t$, $L(t)$ signifies the current position vector of the distinct gorilla, $L_y(t)$ and $L_y(t)$ designates the arbitrary chosen gorilla location, $m_1$, $m_2$, $m_3$, and $m_4$ are arbitrary numbers in the range [0, 1], $T$ is the row vector with elements having value in [-R, R].

Here, $R$, $T$, and $T$ are given by,

$$R = (\cos(2 \times m_A + 1) \times (1 - \frac{t}{max_t})$$  \hspace{1cm} (18)

$$S = R \times v$$  \hspace{1cm} (19)

$$T = Y \times L(t)$$  \hspace{1cm} (20)

Here, max denotes the overall iteration count, $v$ signifies the arbitrary number with value in [-1, 1] and $Y$ depicts the arbitrary value of the problem dimension in [-R, R].

Consider $m_1 \geq 0.5$ and assume $QL(t+1) = L(t+1)$, then Equation (17) can be rephrased as,

$$L(t + 1) = (m_3 - R) \times L(t) + S \times T \times L(t)$$  \hspace{1cm} (21)

The temple solution vector of the SSOA is given by,

$$L_i^{t+1} = L_i^t + stepsz_i$$  \hspace{1cm} (22)

where $stepsz_i$ is given by,

$$Stepsz_i = \chi \times rd(L_d - L_i^t) + \delta \times rd(L_d - L_i^t)$$  \hspace{1cm} (23)

Here, $L_j$, $L_d$, and $L_i^t$ indicates the solution vector of the sheep, horse and shepherd, respectively, $rd$ is an random number with value in between 0 and 1, and the parameters $\chi$ as well as $\delta$ are given as,

$$\chi = X_0 - \frac{X_0}{max_t} \times t$$  \hspace{1cm} (24)
\[ \delta = \delta_0 + \frac{\delta_{\text{max}} - \delta_0}{\text{max}_t} \times t \] (25)

The term \( \delta \) is initialized to \( \delta_0 \) at the starting of the algorithm and increases to \( \delta_{\text{max}} \) as the iteration progresses, \( \chi \) is initialized to \( \chi_0 \) and as the iteration increases, the value reduces to zero.

Now, applying Equation (23) in Equation (22), we get,
\[ L_i^{t+1} = L_i^t + \chi \times rd \delta \left( L_d - L_i^t \right) + \delta \times rd \delta \left( L_j - L_i^t \right) \] (26)

Let us consider, \( L_i^{t+1} = L(t = 1) \) and \( L_i^t = L(t) \), the Equation (26) can be rephrased as,
\[ L(t + 1) = L(t) + \chi \times rd \left( L_d - L(t) \right) + \delta \times rd \delta \left( L_j - L(t) \right) \] (27)
\[ L(t + 1) = L(t) + \chi \times rd \times \left( \tilde{L}_d - \chi \times rd \times L(t) \right) + \delta \times rd \times \left( \tilde{L}_j - \chi \times rd \times L(t) \right) \] (28)
\[ L(t + 1) = L(t) \left( 1 - \chi \times rd - \delta \times rd \right) + \delta \times \left( \chi \times L_d + \delta \times L_j \right) \] (29)
\[ L(t) = \frac{L(t + 1) - \delta \times \left( \chi \times L_d + \delta \times L_j \right)}{1 - \chi \times rd - \delta \times rd} \] (30)

Now, applying Equation (30) in Equation (21), we get,
\[ L(t + 1) = \left( m_3 - R \right) \times L_d(t) + S \times T \times \frac{L(t + 1) - \delta \times \left( \chi \times L_d + \delta \times L_j \right)}{1 - \chi \times rd - \delta \times rd} \] (31)
\[ L(t + 1) = \left( m_3 - R \right) \times L_d(t) + S \times T \times \left( \chi \times L_d + \delta \times L_j \right) \] (32)
\[ L(t + 1) = \left( m_3 - R \right) \times L_d(t) \times \left( 1 - \chi \times rd - \delta \times rd \right) + S \times T \times \left( \chi \times L_d + \delta \times L_j \right) \] (33)
\[ L(t + 1) = \left( m_3 - R \right) \times L_d(t) \times \left( 1 - \chi \times rd - \delta \times rd \right) + S \times T \times \left( \chi \times L_d + \delta \times L_j \right) \] (34)
\[ L(t + 1) = \left( m_3 - R \right) \times L_d(t) \times \left( 1 - \chi \times rd - \delta \times rd \right) + S \times T \times \left( \chi \times L_d + \delta \times L_j \right) \] (35)

wherein, \( L(t+1) \) indicates the position of solution at iteration \( t+1 \), \( L_d \), and \( L_d \) designates the solution vector of the sheep, and horse.

4. Exploitation: while the troop is created by the silverback, it is most powerful and healthy and the remaining members follow the silverback. Eventually, the silverback becomes old and dies, and the younger blackbacks start fighting to establish dominion and mate with the adult females. These patterns are modelled in this phase, wherein follow of silverback and competition for mating is considered. The operation between these two patterns is switched over b considering a parameter \( Z \). When \( R> Z \), the silverback is followed by the other troop members and this is modulated as,
\[ QL(t + 1) = R \times V + \left( L(t) - L_{\text{Silverback}} \right) + L(t) \] (36)

where, \( L(t) \) denotes the present position vector, \( L_{\text{Silverback}} \) indicates the optimal solution attained till now, the term \( R \) is given by Equation (18) and the parameter \( V \) is represented by,
\[ V = \left( \frac{C}{\sum_{i=1}^{C} L_i(t) / C} \right)^{2R} \] (37)

Here, \( C \) designates the population dimension, and \( L(t) \) is the location vector of gorillas in the present iteration. If \( R< Z \), then the blackbacks compete for mating the adult females and the process is modelled as,
\[ QL(t + 1) = L_{\text{Silverback}} - \left( L_{\text{Silverback}} \times Y - L(t) \times Y \right) \times X \] (38)

where,
\[ Y = 2x_m - 1 \] (39)
\[ X = \theta \times I \] (40)
\[ I = \left[ \begin{array}{c} C_1 \quad m_7 \geq 0.5 \\ C_2 \quad m_7 < 0.5 \end{array} \right] \] (41)

Here, \( Y \) denotes the impact force, \( X \) represents the intensity of violence while competing, \( m_6 \) and \( m_7 \) are arbitrary integers in the range \([0, 1]\), \( \theta \) denotes a constant. If \( m_7 \geq 0.5 \), then \( I \) will be an array of size \( 1 \times K \) with random number as elements following normal distribution, else, \( I \) will be a stochastic number following normal distribution, with \( K \) denoting the spatial dimension of the search space. Here, \( L_{\text{Silverback}} \) denotes the best solution computed.

- Fitness re-evaluation: once the location of each gorilla is upgraded, the fitness of each gorilla is re-evaluated to find the best solution.

5. Terminate: the above procedure is kept repeated until the maximalliteration count is attained and the optimal solution is obtained by considering the position of the silverback. The pseudocode of the devised GBSSOA are depicted below in Algorithm (2).

**Algorithm 2: Pseudocode of devised GBSSOA**

*Input*: Population size (\( C \)), Maximal iteration (\( \text{max}_t \))

**Initialization**:  
- Initialize the population of gorillas (\( L(t) \)) randomly with size \( C \).
- Compute the fitness of each gorilla.
- Set iteration counter (\( t \)) to 0.

**Optimization Loop**:
- While \( t < \text{max}_t \)
  - Calculate \( R \).
— Calculate $S$.
— For every gorilla $L_i$, revise the gorilla position.
— Calculate the fitness of every gorilla.
— Keep the best solution as silverback ($L_{Silverback}$).
— For every gorilla $L_i$, if $R \geq Z$ then revise the gorilla position using one equation.
else revise using another equation.
— Revise the optimal solution ($L_{Silverback}$).
— Increment the iteration counter: $t = t + 1$
Output: Best solution ($L_{Silverback}$).

The best solution is given by the position of the silverback $L_{Silverback}$ and by combining the SSOA with GTO; the exploration capability of the GTO is enhanced and thus achieving higher efficiency and achieving effective live VM migration.

5. Results and Discussion

The experimental results of the presented GBSSOA based secure live VM migration is elaborated in this section. The experimental set up, performance metrics, and comparative evaluation of the introduced approach are explained elaborately.

5.1. Experimental Set-Up

The experimentation of the developed GBSSOA based secure live VM migration is carried out by implementing it on a PC with intel core i-3 processor, 2GB RAM, and Windows 10 OS using Java with Cloudsim.

5.2. Evaluation Measures

The effectiveness of the devised GBSSOA based secure live VM migration is inspected in view of different parameters, like workload, QoS, migration time, and trust, and these metrics are shortly briefed below.

1) Workload: workload indicates the overall count of the requests received by the VM from various applications as well as users. The workload of the VM should be always kept low.

2) QoS: QoS parameter has already been elaborated in section 4.2. and is computed using Equation (2).

3) Migration time: migration time is discussed in section 4.2. and is calculated using Equation (1).

4) Trust: the trust parameter is elaborated in section 4.3.2. and is calculated using Equation (8).

5.3. Comparative Techniques

The efficacy of the developed VM migration approach is inspected by comparing it with various schemes, like PSO+GA [30], Kororā framework [7], DC-MFA [25], and PC-TCS [11].

5.4. Comparative Assessment

The comparative evaluation of the devised GBSSOA based secure live VM migration is illustrated in this section. The introduced technique is assessed for its efficacy based on various metrics, like workload, QoS, migration time, and trust based on different task count.

- Assessment with Number of Tasks as 100: in Figure 4, assessment of the devised GBSSOA approach with task size as 100, considering different number of iterations. Figure 4-a) portrays the evaluation of the developed GBSSOA scheme considering workload. The workload computed by the introduced GBSSOA based secure live VM migration is 0.190 with 60 iterations, which is much lower than the value of workload attained by the prevailing methodologies, such as PSO+GA, Kororā framework, DC-MFA, and PC-TCS is 0.224, 0.214, 0.209, and 0.201. In Figure 4 b), the valuation of the GBSSOA system is depicted concerning the QoS parameter. For 70 iterations, the QoS measured by the existing approaches, like PSO+GA, Kororā framework, DC-MFA, and PC-TCS and the devised GBSSOA approach is 0.441, 0.469, 0.475, 0.485, and 0.499, respectively. The devised technique is revealed to have attained a higher value of QoS, thus showing improved performance. Figure 4-c) depicts the evaluation of the proposed GBSSOA technique with regard to migration time. The proposed GBSSOA approach measured migration time of 389.658ms, which is lower that the migration time of the traditional methods, such as PSO+GA, Kororā framework, DC-MFA, and PC-TCS corresponding to 472.975ms, 432.259ms, 420.258ms, and 410.854ms, with 80 iterations. Figure 4-d) depicts the valuation of the presented GBSSOA technique in view of the trust. The trust achieved by the conventional approaches, like PSO+GA, Kororā framework, DC-MFA, and PC-TCS, and the presented GBSSOA is 0.433, 0.451, 0.466, 0.488 and 0.505, respectively with 50 iterations.

![Figure 4. Valuation of the introduced GBSSOA based secure live VM migration with task count as 100.](image-url)
• Assessment with Number of Tasks as 200: the evaluation of the developed GBSSOA based VM migration is carried out with 200 tasks, considering various metrics and this is presented in Figure 5. In Figure 5-a), the workload-based analysis of the GBSSOA scheme. The workload measured by the proposed GBSSOA for 70 iterations is 0.166. On the other hand, the existing techniques, like PSO+GA, Kororā framework, DC-MFA, and PC-TCS had higher workload of 0.199, 0.192, 0.185, and 0.179, correspondingly. Figure 5-b) portrays the assessment of the introduced GBSSOA method with respect to QoS. The conventional methods, like PSO+GA, Kororā framework, DC-MFA, and PC-TCS achieved a low value of QoS at 0.485, 0.514, 0.533, and 0.566 with 80 iterations, when compared to the QoS of 0.587 measured by the developed approach. The evaluation of the proposed GBSSOA based secure live VM migration based on migration time is illustrated in Figure 5-c). Considering a total of 50 iterations, the developed GBSSOA and the prevailing schemes, like PSO+GA, Kororā framework, DC-MFA, and PC-TCS required migration time of 310.258ms, 382.587ms, 370.254ms, 361.855ms, and 352.875ms, respectively. In Figure 5-d), the evaluation of the devised secure live VM migration is displayed. The developed GBSSOA based live VM migration attained a trust value of 0.515, while the trust value measured by the approaches, 0.459 for PSO+GA, 0.476 for Kororā framework, 0.488 for DC-MFA, and 0.499 for PC-TCS, with 60 iterations.

5.5. Comparative Discussion

This section details the comparative evaluation of the proposed GBSSOA based secure live VM migration scheme. The efficacy of the devised technique is investigated considering different parameters, like workload, QoS, and migration time considering different number of tasks by varying the iteration count.
Table 2. Comparative discussion of the introduced GBSSOA based secure live VM migration scheme.

<table>
<thead>
<tr>
<th>Number of tasks</th>
<th>Metrics</th>
<th>PSO+GA</th>
<th>Kororā framework</th>
<th>DC-MFA</th>
<th>PC-TCS</th>
<th>GBSSOA-based secure live VM migration</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Workload</td>
<td>0.207</td>
<td>0.190</td>
<td>0.185</td>
<td>0.175</td>
<td>0.169</td>
</tr>
<tr>
<td></td>
<td>QoS</td>
<td>0.485</td>
<td>0.499</td>
<td>0.533</td>
<td>0.554</td>
<td>0.578</td>
</tr>
<tr>
<td></td>
<td>Migration time (ms)</td>
<td>479.954</td>
<td>442.253</td>
<td>423.256</td>
<td>412.258</td>
<td>392.256</td>
</tr>
<tr>
<td></td>
<td>Trust</td>
<td>0.486</td>
<td>0.499</td>
<td>0.501</td>
<td>0.521</td>
<td>0.541</td>
</tr>
<tr>
<td>200</td>
<td>Workload</td>
<td>0.180</td>
<td>0.179</td>
<td>0.171</td>
<td>0.169</td>
<td>0.161</td>
</tr>
<tr>
<td></td>
<td>QoS</td>
<td>0.497</td>
<td>0.533</td>
<td>0.556</td>
<td>0.587</td>
<td>0.599</td>
</tr>
<tr>
<td></td>
<td>Migration time (ms)</td>
<td>445.854</td>
<td>425.258</td>
<td>402.854</td>
<td>392.255</td>
<td>392.254</td>
</tr>
<tr>
<td></td>
<td>Trust</td>
<td>0.487</td>
<td>0.510</td>
<td>0.525</td>
<td>0.541</td>
<td>0.554</td>
</tr>
<tr>
<td>300</td>
<td>Workload</td>
<td>0.199</td>
<td>0.195</td>
<td>0.169</td>
<td>0.161</td>
<td>0.141</td>
</tr>
<tr>
<td></td>
<td>QoS</td>
<td>0.541</td>
<td>0.587</td>
<td>0.614</td>
<td>0.633</td>
<td>0.654</td>
</tr>
<tr>
<td></td>
<td>Migration time (ms)</td>
<td>402.258</td>
<td>385.541</td>
<td>368.541</td>
<td>365.854</td>
<td>355.254</td>
</tr>
<tr>
<td></td>
<td>Trust</td>
<td>0.488</td>
<td>0.514</td>
<td>0.539</td>
<td>0.559</td>
<td>0.569</td>
</tr>
</tbody>
</table>

Table 2 depicts the comparative discussion of the introduced GBSSOA based secure live VM migration and the values are portrayed corresponding to the iteration count of 90. It can be deduced from the table that the devised GBSSOA scheme achieved a minimal value of workload and migration time of 0.141, and 342.254ms, and a maximal QoS of 0.654 and maximum trust of 0.569. The high value of QoS and trust are achieved owing to the utilization of the presented GBSSOA technique for the optimization of the migration process. Further, minimal workload and migration time is ensured by the allocation of the VM in a round robin manner. Thus, compared to existing approaches, GBSSOA exhibits both lower workload and superior QoS during live VM migration. Notably, it achieves the fastest migration times and highest trust values, indicating efficient resource utilization and reliable service guarantees. This suggests GBSSOA’s potential as a leading method for high-performance live VM migration.

6. Conclusions

Live VM migration is the process of migrating an operational VM from a server to other servers by ensuring that the application services are provided uninterruptedly during the migration process thereby ensuring QoS. A new method for providing security during the process of live VM migration is developed using hybrid optimization in this paper. Initially, the tasks are assigned to the VM using a round robin manner. After task allocation, the workload on the PM is computed based on migration time and QoS. The computed workload is equated with the threshold to determine if VM migration is needed. If the workload is greater than the threshold, live VM migration is performed using the introduced GBSSOA, based on fitness objectives, such as run time of the task, migration time and QoS. Moreover, the security of the VM migration is ensured by considering the trust value along with the above-mentioned parameters. The developed GBSSOA-based secure live VM migration is evaluated using measures, such as workload, QoS, migration time, and trust and is revealed to have achieved values of 0.141, 0.654, 342.254ms, and 0.569 correspondingly. Future direction of this research comprises inclusion of other objectives, such as cost and power consumption for enhancing the migration process.

References


[28] Xie X., Yuan T., Zhou X., and Cheng X.,


Gokul Narayanan pursuing PhD in Computer Science and Engineering from Vellore Institute of Technology, Vellore, India under the guidance of Dr. RA. K Saravanaguru and specialized in Cloud Computing, Information Security and Quality Assurance.

Saravanaguru Kannan a seasoned professor in the School of Computer Science and Engineering (SCOPE) at Vellore Institute of Technology. Former Assistant Dean of Academics with 23 years of expertise in teaching and research. He’s published in prestigious journals with IEEE, ACM, Springer, and Elsevier publishers. Engaged in cutting-edge projects like a context-aware middleware for web services, a blockchain-based platform for smart contracts, and AI/ML applications in intelligent vehicle technology. Driven to collaborate with motivated and talented students. Reach out via email at saravanank@vit.ac.in https://www.linkedin.com/in/saravanagururak.