Secured Multi-Objective Optimization for Virtual Machine Placement in the Cloud Security

Rajkumar P[1], Dr.G.Thippanna[2], Dr.A.Pratapa Reddy[3]

[1]Research Scholar Department of Computer Science and Engineering Niilm University Haryana.rajmatrix2000@gmail.com
[2]Associate professor Department of Computer Science and Engineering Niilm University Haryana gt.pana2012@gmail.com
[3]Co- Supervisor, Associate professor, Department of Computer Science and Engineering,SVS Group of Institutions.

This study explores the Virtual Machine Placement (VMP) optimization problem and its objectives, focusing on maximizing resource utilization, minimizing energy consumption, reducing network latency, and ensuring high availability. In response, the Secured Multi-Objective Optimization based Virtual Machine Placement algorithm (SMOOP) is proposed and evaluated. SMOOP aims to simultaneously optimize multiple criteria while meeting security requirements. The algorithm's effectiveness is assessed through rigorous evaluation, offering promising insights into efficient and secure virtual machine placement strategies.
Keywords: SMOOP, VMs, Machine Learning Techniques.

1. Introduction

The formulation of the VMP (Virtual Machine Placement) optimization problem typically revolves around allocating virtual machines to physical servers in a way that maximizes resource utilization, minimizes energy consumption, reduces network latency, and ensures high availability and reliability of services. The objectives we aim to accomplish with VMP optimization include:

- Resource Utilization Maximization: Efficiently allocate virtual machines to physical servers to maximize the utilization of CPU, memory, storage, and other resources, thus reducing wasted capacity.
- Energy Consumption Minimization: Consolidate virtual machines onto a minimal number of physical servers to reduce overall energy consumption and operational costs.

- Network Latency Reduction: Place virtual machines in such a way that communication between them experiences minimal latency, ensuring optimal performance for applications and services.
- High Availability and Reliability: Distribute virtual machines across physical servers to mitigate the risk of single points of failure and ensure continuous availability of services in case of hardware failures or maintenance.

Secured Multi-Objective Optimization based Virtual Machine Placement (SMOOP) algorithm addresses these objectives by simultaneously optimizing multiple criteria, such as resource utilization, energy consumption, and reliability, while ensuring security requirements are met. The formulation of SMOOP involves defining the following:

- Objective Functions: These functions quantify the goals we want to achieve, such as maximizing resource utilization, minimizing energy consumption, reducing network latency, and enhancing reliability.
- Constraints: These are the conditions that must be satisfied during the placement process. Constraints may include security requirements, hardware limitations, and application-specific considerations.
- Decision Variables: These are the variables that the optimization algorithm manipulates to find the optimal placement solution. They represent decisions such as which virtual machines to place on which physical servers.
- Optimization Algorithm: SMOOP utilizes a multi-objective optimization algorithm to search for the optimal solution that balances the trade-offs between different objectives. This algorithm iteratively explores the solution space to find a set of Pareto-optimal solutions, where no solution can be improved in one objective without worsening at least one other objective.

By formulating the VMP problem and objectives in this way, SMOOP aims to provide efficient, secure, and balanced virtual machine placement solutions that meet the diverse needs of modern data center environments.

2. SMOOP Design

With the proposed security metric for VMs, we can quantify the risk level of a cloud. As a typical multi-objective optimization problem, the objectives may conflict with each other. For instance, placing more VMs on a physical server can reduce resource wastage and network traffic but also increase security risks due to the co-residency problem. It is impractical to always find a solution that minimizes all objectives simultaneously.

Evolutionary Multi-Objective Algorithms (EMOAs), such as NSGA-II, are popular solutions for these types of problems. Using EMOAs, we can obtain Pareto-optimal solutions that balance the objectives of security, network traffic, and resource utilization. These solutions provide a set of optimal trade-offs, allowing us to effectively manage the competing objectives in cloud VM placement.

It's important to note that our goal is to enhance the survivability of the entire cloud, not to optimize the solution for any specific VM. As a result, while our approach may lower the security level of a particular VM, the overall security level of the cloud can still be improved.

Algorithm:SMOOP

```
# Initialize the candidate population using some strategy
Candidate = init_by_strategies()
# Loop over the number of generations
for G in range(1, N_G + 1):
        # Elite selection phase
    Elite = [ ]
    for i in range(1, N_E + 1):
        Elite.append (Elite_choosing (Candidate))
        # Crossover phase to create offspring
    Off_ C = [ ]
    for j in range(1, N_C + 1):
        X, Y = Random_ select(Candidate)
        Off_ C.append (Crossover(X, Y))
        # Mutation phase to create mutated offspring
    Off_M = [ ]
    for k in range(1, N_M + 1):
        X = Random_ select(Candidate)
        Off_M.append (Mutation(X))
        # Combine Elite, crossover offspring, and mutated offspring
    Temp = Elite + Off_C + Off_M
        # Sort the combined population by fitness
    Temp = fitness_sorting(Temp)
        # Select the top candidates to form the new candidate population
    for i in range(1, N_G + 1):
        Candidate[i] = Temp[i]
```

3. Implementation and Evaluation of SMOOP in Java

We implemented our solution in Java and conducted all experiments in a simulation environment. The input data for these experiments is provided through configuration files, and we employed multiple threads to enhance performance.

3.1.Experimental Setup

1. Random VM Generation:

   - A large number of VMs with diverse parameters are randomly generated to evaluate SMOOP.
   - Each VM is deployed onto a physical machine without considering migration costs during the initial placement.

2. VM Parameters:

   - For each VM, requirements for CPU, memory, and disk space are randomly assigned.
   - Vulnerability scores are assigned based on a uniform distribution.

3. Physical Machine Configuration:

   - Physical machines are configured similarly to VMs, with randomly assigned parameters.
   - The numbers of VMs and physical machines vary across different experiments to assess performance under various conditions.

3.2. Performance Optimization

Multithreading: The use of multiple threads enhances performance and efficiency, enabling effective management of large-scale simulations. This experimental setup is designed to comprehensively evaluate SMOOP's effectiveness and efficiency under diverse conditions and configurations..

3.3.Computing Complexity

Assume there are M physical servers and N virtual machines (VMs). Our algorithm iterates k times, with a candidate pool size of P in each iteration. Here, N is significantly larger than the other factors. The objective values for each Virtual Machine Placement (VMP) can be computed within a bound of O(MN^2). The fitness value of a VMP is calculated in constant time using a fitness function, resulting in a complexity of O(P) for the entire candidate pool. Sorting the candidate pool has a complexity of O(P log P) . The elite selection operation takes constant time, and the crossover and mutation operations are bounded by O(MN^2) . Thus, the overall combined complexity of our algorithm is O(k(MN^2 + PMN^2 + P log P)) , simplifying to O(kPMN^2).

If we consider the network's cascade effect, the computation time for the risk value of each VM changes from O(N^2) to O(N^3) , due to constructing all possible attack paths using Depth-First Search. Consequently, the algorithm's complexity would be bounded by O(kPN^3) .

To improve efficiency, we simplified the risk value computation while preserving correct correlations. Additionally, we employed multithreading in our implementation, using 8 threads to simultaneously conduct elite selection, crossover, and mutation operations.

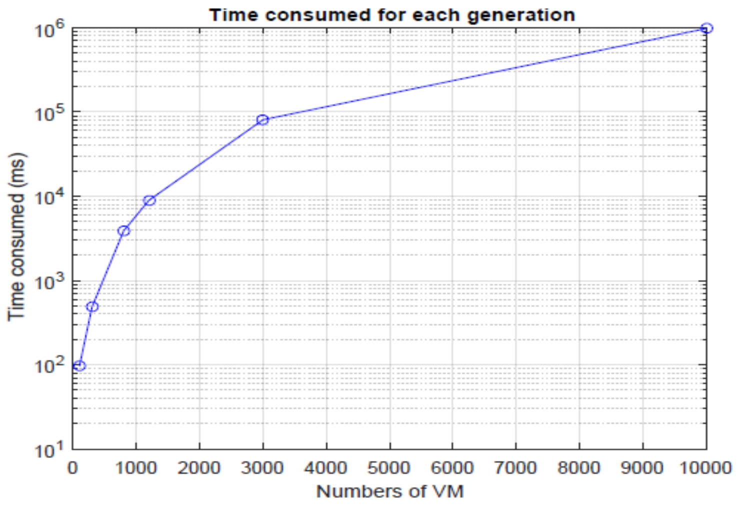

Figure 1: Scalability

We tested our implementation on an 8-core processor with 16GB of memory. The performance of our algorithm depends on the number of VMs, the number of physical machines, and the number of candidate placements generated in each generation. Figure 1 shows the computation time per generation under the following settings: 1. 100 different placements are generated for each generation. 2. Each generation involves 270 operations. With 10,000 VMs and 500 physical machines, each generation takes about 15-20 minutes. When the number of VMs is reduced to 3,000, each generation takes approximately 2 minutes.

4. Effectiveness in Risk Reduction

Security risk is a crucial factor in VMP. To assess whether our strategies can enhance the overall security of the cloud, we conducted experiments focusing solely on risk level during placement. Figure 2 demonstrates that non-secure network traffic bandwidth is also reduced by approximately 3% to 5% when optimizing multiple objectives.

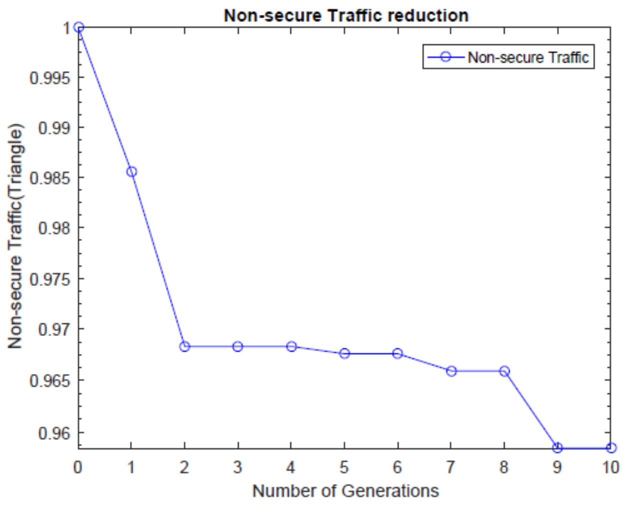

Figure 2: Non-secure Network Traffic

4.1.Effectiveness in Workload Balance

In our algorithm, we prioritize evenly distributing the workload among active physical machines once their number is determined. Figure 3 presents experimental results with the following weight settings: 0.4 for risk level, 0.4 for workload balance, 0.1 for resource wastage, and 0.1 for network traffic in an environment consisting of 400 VMs and 60 physical machines. According to our experimental setup, the memory request of a VM ranges from 4GB to 40GB, and each physical machine can provide 256GB of memory.

4.2.Effectiveness of Multi-Objective Optimization

In this evaluation, we consider multi-objectives on risk level, resource wastage, and network traffic. Figure 4 shows experimental results with weight setting (0.8 (risk level), 0.1 (Resource Wastage), 0.1 (Network Traffic)) in an environment of 800 VMs and 60 physical machines. The risk level has weight of 80%, resource wastage and network traffic have weight of 10% for each in the fitness function. We collect the placement with the best fitness value. The baseline is still the best placement chosen from 100 random-FFD placements. If a physical machine can hold hundreds of VMs, the placement generated by FFD will be using the minimum number of physical With a weight setting of (0.8, 0.1, 0.1), the number of active physical machines is constrained, and there's notable reduction in resource wastage, leading to enhanced security measures.

In another experiment, employing weight settings of (0.4 for risk level, 0.3 for Resource Wastage, and 0.3 for Network Traffic) within an environment hosting 3000 VMs and 200 physical machines, the results are depicted in Figure 5. Given the higher weights assigned to resource wastage and network traffic, there's a controlled allowance for resource wastage, consequently impacting the level of security improvement achievable. It's worth noting that a cloud provider can always adjust optimization preferences by modifying the weights assigned to different objectives.

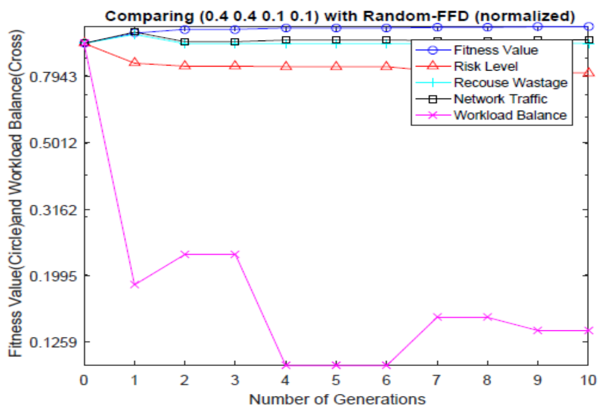

Figure 3: Multi-objective optimization with Workload balance

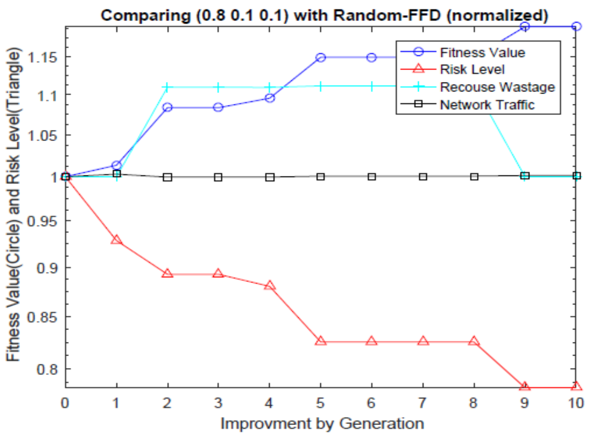

Figure 4: Multi-objective optimization

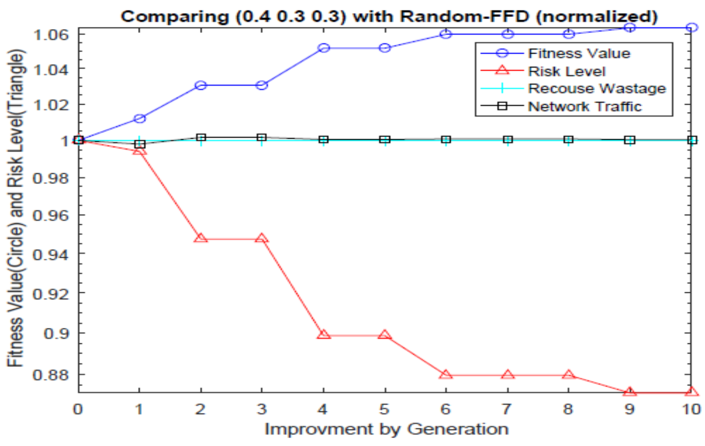

Figure 5: Multi-objective optimization 2

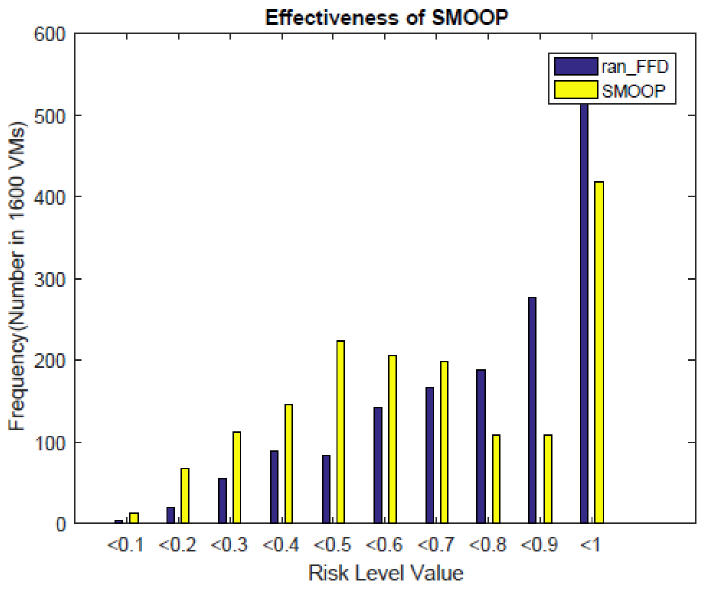

Figure 6: Comparison with Distribution in 1600 VMs and 120 PMs

5. Comparison with random-FFD algorithm

In our experiment with 1600 VMs and 120 physical machines, we generated 100 placements using the random-FFD algorithm. We selected the placement with the lowest median risk level value and then applied our algorithm to further reduce this risk level, ultimately choosing the best placement.

As depicted in Figures 6 and 7, there's a noticeable decrease in the overall risk level of the VM set. The X-axis represents the risk level values of VMs, with intervals such as "<0.2" indicating a risk level between 10% and 20%. With 1600 VMs, the proportion of VMs with a risk level under 50% saw an improvement ranging from 15% to 35%. Additionally, the percentage of VMs with a risk level above 80% decreased from 54% to 33%. These results illustrate how our placement strategies significantly enhance the security level of the entire cloud infrastructure.

These experimental outcomes underscore the effectiveness of our approach and its superiority over existing solutions. However, it's essential to note that while we quantified the security risk of the cloud, it remains a coarse-grained relative value. Although the median value proved effective, it may not always represent the optimal choice. Moreover, as discussed, the prioritization of strategies can be dynamically adjusted in response to real-time environmental conditions, and new strategies may need to be incorporated to adapt to evolving threats. This adaptive capability ensures that our system continues to evolve to effectively address real-world challenges.

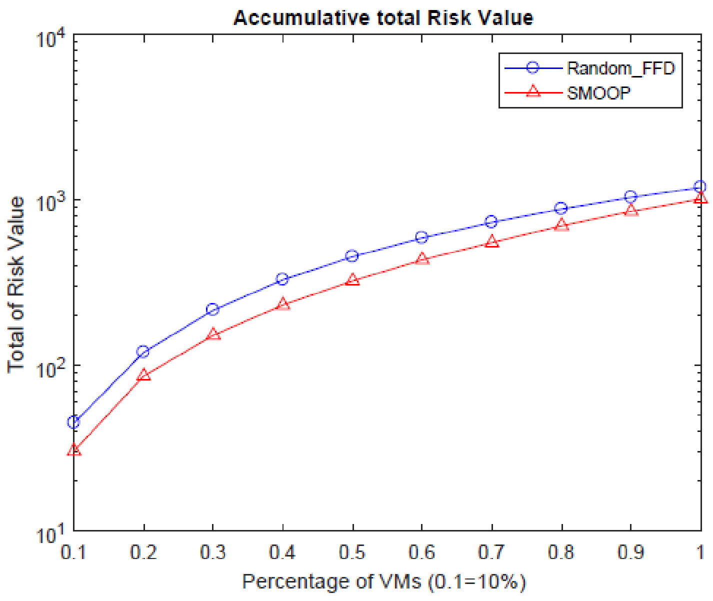

Figure 7: Accumulative Risk Value

6. Conclusion:

A comprehensive approach to security assessment in virtual machine placement, addressing vulnerabilities stemming from various factors such as networks, physical machines, VMs, and co-residence of VMs. Chapters 3 and 4 detail your security metrics, providing a solid foundation for measuring the security risks associated with specific VM placements. Your innovative approach incorporates multiple objective optimization to generate improved VM placements while considering resource constraints, demonstrating its effectiveness through experimental validation.

References

1 George Amvrosiadis, Jun Woo Park, Gregory R. Ganger, Garth A. Gibson, Elisabeth Baseman, and Nathan DeBardeleben. On the diversity of cluster workloads and its impact on research results. In Proceedings of the 2018 USENIX Conference on Usenix Annual Technical Conference, USENIX ATC ’18, pages 533–546, Berkeley, CA, USA, 2018. USENIX Association.

2 M. Aslam, C. Gehrmann, and M. BjÃfjrkman. Security and trust preserving vm migrations in public clouds. In 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications, pages 869–876, June 2012.

3 Yossi Azar, Seny Kamara, Ishai Menache, Mariana Raykova, and Bruce Shepard. Co-location-resistant clouds. In Proceedings of the 6th Edition of the ACM Workshop on Cloud Computing Security, CCSW ’14, pages 9–20, New York, NY, USA, 2014. ACM.

4 Khalid Bijon, Ram Krishnan, and Ravi Sandhu. Mitigating multi-tenancy risks in iaas cloud

through constraints-driven virtual resource scheduling. In Proceedings of the 20th ACM Symposium on Access Control Models and Technologies, SACMAT '15, pages 63–74, New York, NY, USA, 2015. ACM.

5 Eli Cortez, Anand Bonde, Alexandre Muzio, Mark Russinovich, Marcus Fontoura, and Ricardo Bianchini. Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms. In Proceedings of the 26th Symposium on Operating Systems Principles, SOSP '17, pages 153–167, New York, NY, USA, 2017. ACM.

6 D. Gonzales, J. M. Kaplan, E. Saltzman, Z. Winkelman, and D. Woods. Cloud-trust-a security assessment model for infrastructure as a service (iaas) clouds. IEEE Transactions on Cloud Computing, 5(3):523–536, July 2017.

7 Apruzzese G., Colajanni M., Ferretti L., Marchetti M. (2019). "Addressing adversarial attacks against security systems based on machine learning," in 2019 11th International conference on cyber conflict (CyCon), Tallinn, Estonia, May 28–31, 2019 (IEEE;), 900, 1–18

8 Hesamifard E., Takabi H., Ghasemi M., Jones C. (2017). "Privacy-preserving machine learning in cloud," in Proceedings of the 2017 on cloud computing security workshop, 39–43