International Journal of Advance Research in Engineering, Science & Technology(IJAREST),
ISSN(O):2393-9877, ISSN(P): 2394-2444,
Volume 2, Issue 5, May- 2015, Impact Factor: 2.125

Efficient and Secure Allocation of Virtual Machines Using Resource Allocation Policies in Cloud Computing

Hina Patel[1], Dr. Hiren B. Patel[2]

[1] Computer engineering, S.k. Patel College of engineering, Visnagar, Gujarat, India,hinapatel91@gmail.com
[2] Computer engineering, S.k. Patel College of engineering, Visnagar, Gujarat, India, hbpatel1976@yahoo.com

Abstract

Apart from providing numerous benefits such as On-demand self-service, Broad network access, Resource pooling, Rapid elasticity, Measured service, Scalability, Availability, Secure access etc. through virtualization, Cloud computing introduces a number of new security risks. Virtualization means multiple VM machine on single physical machine. In Cloud computing allocation of virtual machine is an issue that has been addressed in many computing area such as operating systems, grid computing, and datacenter management. The huge amount of data on Cloud naturally becomes targets for attackers. We aim to address one of such attacks, co-residence or co-location attack, where malicious users may co-locate their infected virtual machines(VMs) with the normal working VMs on the same server. Using this, attackers may sneak into user's confidential data using side channel attack. In our research, we intend to achieve efficient allocation of virtual machine in a secure way for Cloud environment. In connection to this, we try to minimize the probability of VM co-location which aim to results into improvement of coverage rate.

Keywords- *allocation policy, Cloud computing, Coverage rate virtual machine allocation, Virtualization*

I. INTRODUCTION

Cloud computing is model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources(e.g. networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. The Cloud model is composed of five essential characteristics (on-demand self-service, broad network address, resource pooling, rapid elasticity, measured service), three service models (software as a service, platform as a service, infrastructure as a service), and four deployment models (public, private, community and hybrid). [1]

The appearance of Cloud computing has superficially changed the use of information technologies. More and more IT resources, like software applications, operating systems, and even network infrastructure, are now being delivered as services and made accessible to a wide range of customers. On the other hand, malicious users are also targeting the growing amount of data in Cloud platforms, which creates a major potential security risk. This paper focuses on a novel type of threat: the co-resident attack [2] (also known as a co-residence attack or co-location attack). In Cloud computing, to maximize the utilization rate of hardware platforms, it is common observation that the virtual machines (VMs) of different users run on the same physical server (i.e., these VMs are co-resident), and are logically isolated from each other. However, malicious users can avoid the logical isolation, and obtain sensitive information from co-resident VMs [2]. If Cloud providers cannot ensure data confidentiality and hence lose the basic trust from users, the future of Cloud computing will be jeopardize. As a result, it is important to find effective and practical countermeasures against this kind of threat. While, in principle, programs running on co-resident VMs should not be able to manipulate each other. There are a variety of ways this can occur in practice. For example, the cache utilization rate has a major influence on the execution time of cache read operations. Therefore, the attacker is able to estimate the victim's cache usage by performing extensive cache read operations and comparing the execution time on a co-resident VM [3]. With similar approaches, attackers can also infer other private statistics, such as the traffic rate of a website. In addition, co-resident VMs shares the instruction cache and other hardware resources. This can also be exploited by malicious users to extract private information, such as cryptographic keys [3], although it requires overcoming several major challenges. Furthermore, a number of papers have discussed how to build side channels between co-resident VMs to transfer sensitive information, which is prohibited by security policies [4-9]. If we can find a practical way to decrease the possibility of achieving co-location, then the threat of this kind of attack can be mitigated. Specifically, we focus on the VM allocation policy, since this is one important factor the Cloud provider can control that will influence the possibility of co-location.

II. BACKGROUND THEORY

Normally, VMs are placed on a same physical server for better resource utilization. Although VMs co-reside with each other, they are logically isolated. Malicious users may break this isolation and try to access of user's confidential data. This logical isolation may be broken using side channel attack. To eliminate side channel attack, researchers propose methods,

which result into substantial changes in existing system, and hence, may not result into deployment. So it is required to propose a solution where with a minimum or no changes at underlying layer, the same can be practically deployable. To achieve efficiency & secure allocation of virtual machine in Cloud enviroment, we plan to implement a security game model to minimize the attacker's possibility of achieving co-location by selecting a policy with certain probability, from a common pool of policies. This will not require any changes to the existing system & hence practically deployable.

III. RELATED WORK

The problem of VM allocation can be divided into two parts:

(a) The first part is admission of new requests for VM provisioning and placing the VMs on hosts
(b) Whereas the second part is optimization of current allocation of VMs [10].

To address these issues various policies are used in VM allocation.

1. **Selection Policy:**

The selection policy decides which process has to be migrated when a host node is imbalanced. Selection policy selects random node from all this node. [11].

2. **Transfer Policy**

The Transfer policy performs one of the very important tasks of deciding a virtual machine of a job which is authorized, or not to compete in the process of migration. File transfer is a critical component in the application of Cloud computing, moreover, efficient and flexible file transfer with reliability has an important role in guaranteeing a good quality of service for users [11].The policy offers benefit of transfer of large enterprise files in a secure and compliant manner and secure access from any device or location using a browser, with or without VPN.

3. **Location Policy**

The location policy also perform important task of finding a suitable server where a virtual machine can be migrated without making it overloaded. The main work of this step is to find such a host machine that is nearly as much lighter as the local host node is heavier compared to the overall average, which results in both host nodes balanced after the migration [16]. The policy offers benefit of it gives high performance on distributed load balancing. Finds a good lightly-loaded node that minimizes useless polling and maximizes even load distribution. The problem with this policy is, it is select node randomly.

4. **Information Policy**

The information policy works to collect the information about the host nodes and also decides when information about the host nodes in the system is collected. It follow a periodic policy, having each host node broadcasting its load information periodically to all host nodes in the cluster, which also serves as a heartbeat denoting a message to the host presence in the system[11]. The policy offers benefit of it gives all the information of host node. The problem with this policy is, there are some issues while using information polices which are the interaction between human beings and technology for using information.

5. **Round Robin Policy**

Round Robin policy is simple. For each new virtual machine, it repeat sequentially through available hosts until it finds one that has sufficient free resources available to host the virtual machine. Once, it matches the virtual machine to that host. For the next virtual machine, the policy repeat through the hosts sequentially, starting where it left off, and again choosing the first host that can serve the virtual machine. This process is continued until all virtual machines are allocated. It is currently the default scheduling policy in the Eucalyptus Cloud platform [12, 13]. The policy offers benefit of it gives better result for small processes. The problem with this policy is it does not work for time series based processes.

6. **Striping Policy**

In Striping scheduling policy each new virtual machine, it first eliminate all hosts that do not have the available resources to host the virtual machine. From the left hosts, it finds the one that is currently hosting the least number of virtual machines. Once, it matches the virtual machine to that host. This process is repeated until all virtual machines are allocated. It is currently available as one of the built-in policies in the Open Nebula Cloud platform [12, 14]. The policy offers benefit of the striping policy evenly distributes the number of virtual machines across the host systems. The problem with this policy is some characters and strips use highly unconventional methods of communication

7. **Packing Policy**

The Packing policy is the adverse of the Striping policy. For each new virtual machine, it first discards all hosts that do not have the available resources to host the virtual machine. From the remaining hosts, it finds the one that is currently hosting the greatest number of virtual machines. Once, it matches the virtual machine to that host. This process is repeat until all virtual machines are allocated. It is currently available as one of the built-in policies in the Open Nebula Cloud platform, and implemented as the Greedy policy option in Eucalyptus [12, 14]. The policy offers benefit minimize the number of system

data stores in use Pack the VMs in the system data stores to reduce VM fragmentation. The problem with this policy is using packing policy; we create Windows and/or Mac OS packages (MSI or PKG files) and then using any third-party deployment tool that supports the deployment of native installers

8. **Load Balancing (free CPU count) Policy**

The count-based Load Balancing policy is a more growing version of the Striping policy. For every new virtual machine, it first rejects all hosts that do not have the available resources to host the virtual machine. From the remaining hosts, it finds the one that with the greatest number of free CPU cores. Once found, it matches the virtual machine to that host. This process is repeated until all virtual machines are allocated. The Load Balancing policy minimizes the CPU load on the hosts. It is currently available as one of the built-in policies in the Open Nebula Cloud platform as the Load Aware policy [12]. The policy offers benefit, it minimizes the CPU load on the host. The problem with this policy is it uses probability approach to ease the problem of load balancing in the concurrent user's scene.

9. **Load Balancing (free CPU ratio) Policy**

The ratio-based Load Balancing policy is a more advanced version of the count-based Load Balancing policy. For every new virtual machine, it first rejects all hosts that do not have the available resources to host the virtual machine. From the remaining hosts, it matches the one that with the greatest ratio of free CPU cores to allocated CPU cores. Once, it matches the virtual machine to that host. This process is continued until all virtual machines are allocated. The Load Balancing policy minimizes the CPU load on the hosts and is designed to do a slightly better job than the count-based Load Balancing policy [13].

10. **Watts per Core Policy**

The Watts per Core policy is energy-saving policy. For every new virtual machine, it first rejects all hosts that do not have the available resources to host the virtual machine. From the remaining hosts, it detect the one that would result in using the least additional wattage per CPU core if chosen, based on each host's power supply. Once, it matches the virtual machine to that host. This process is repeat until all virtual machines are allocated .The Watts per Core policy endeavor to always find the host that will take up the least additional wattage per core, reducing overall energy consumption [13].It always find the host that will take up the least additional wattage per core, it reduce overall energy consumption.

11. **Cost per Core Policy**

The Cost per Core policy is intends cost-saving policy. For every new virtual machine, it first rejects all hosts that do not have the available resources to host the virtual machine. From the remaining hosts, it finds the one that would result in using the least additional cost per CPU core if chosen, based on each host's power supply and electricity costs. Once, it matches the virtual machine to that host. The Cost per Core policy makes the same assuming that the Watts per Core policy does. This process is repeated until all virtual machines are allocated. The Cost per Core policy attempts to always find the host that will take up the least additional cost per core, reducing overall energy costs [13]. The policy offers benefit of the Cost per core policy is energy-aware policy and also minimizes the cost estimation by seeking the host that would capture least additional cost per core.

12. **Least VM/Most VM Policy**

For each new VM request, the policy will select the server that hosts the least/most number of VMs, among those with sufficient resources left. This kind of policy expands the workload within the system for better workload balance/lower energy consumption [15].

13. **Random Policy**

For each new VM request, the policy will randomly choose one server from those having enough resources [15].

14. **Control Policy**

We use proportional thresholding for the control policy of the HSC (Horizontal Scale Controller). We fit a function to empirical measurements of the CPU utilization of data nodes (storage nodes) at various load levels to determine the parameter values to use for proportional thresholding [16].

Based on the literature survey presented in previous section, we have identified coverage rate and efficiency as a key parameter area of improvement in the process of VM allocation process. In this section, we propose a approach which aim to handle coverage rate and efficiency.

IV. PROPOSED WORK

We aim to address one of such attacks, co-residence or co-location attack, where malicious users may co-locate their infected virtual machines (VMs) with the normal working VMs on the same server. Using this, attackers may sneak into user's confidential data using side channel attack. In our research, we intend to achieve efficient allocation of virtual machine in a secure way for Cloud environment. In connection to this, we try to minimize the probability of VM co-location which results into improvement of coverage rate.

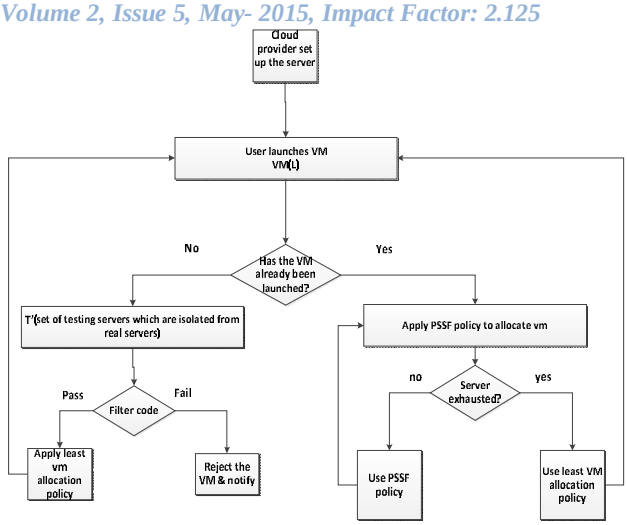

Fig-1-Proposed System Flowchart

V. EXPERIMENTAL SETUP

In this section, we present a comparison between the results predicted by our models, and the results calculated from simulation experiments, Efficiency and Coverage. In order to verify the above models of Efficiency and Coverage; we conducted our simulation experiments on the widely used platform Cloudsim [17, 18].

VI. CONCLUSION & FUTURE WORK

Co-resident attack is a major threat to data privacy in Cloud computing. In our work, we evaluate three basic VM allocation policies from a security perspective. We discover that the Most VM policy performs the best. In addition, we aim to propose a new policy that claims to increases the difficulty for attackers to achieve co-residence. We further intend to test our findings on practical environment.

VI. REFERENCES

[1] **NIST** SP 800-145, "**A NIST definition of Cloud computing**", http://csrc.nist.gov/publications/drafts/800-145/Draft-SP-800 145_Cloud-definition.pdf.

[2] Ristenpart, T., Tromer, E., Shacham, H., and Savage, S.: "**Hey, You, Get Off of My Cloud: Exploring Information Leakage in Third-Party Compute Clouds,**" Proc. 16th ACM Conference on Computer and Communications Security (CCS 2009), 2009, pp. 199-212.

[3] Zhang, Y., Juels, A., Reiter, M., and Ristenpart, T.: "**Cross-VM Side Channels and Their Use to Extract Private Keys,**" Proc. 2012 ACM Conference on Computer and Communications Security - CCS '12, 2012, pp. 305-316.

[4] Okamura, K., Okamura, K., and Oyama, Y.: "**Load-based Covert Channels between Xen Virtual Machines**," Proc. 2010 ACM Symposium on Applied Computing - SAC '10, 2010, pp. 173-180.

[5] Hlavacs, H., Treutner, T., Gelas, J.-P., Lefevre, L., and Orgerie, A. - C.: "**Energy Consumption Side-Channel Attack at Virtual Machines in a Cloud,**" Proc. 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing, 2011, pp. 605-612.

[6] Wu, J., Ding, L., Wang, Y., and Han, W.: "**Identification and Evaluation of Sharing Memory Covert Timing Channel in Xen Virtual Machines,**" Proc. 2011 IEEE 4th International Conference on Cloud Computing, 2011, pp. 283-291.

[7] Xu, Y., Bailey, M., Jahanian, F., Joshi, K., Hiltunen, M., and Schlichting, R.: "**An Exploration of L2 Cache Covert Channels in Virtualized Environments,**" Proc. 3rd ACM Workshop on Cloud Computing Security - CCSW '11, 2011, pp. 29-39.

[8] Kadloor, S., Kadloor, S., Kiyavash, N., and Venkitasubramaniam, P.: "**Scheduling with Privacy Constraints,**" Proc. 2012 IEEE Information Theory Workshop, 2012, pp. 40-44.

[9] Xia, Y., Yetian, X., Xiaochao, Z., Lihong, Y., Li, P., and Jianhua, L.: "**Constructing the On/Off Covert Channel on Xen,**" Proc. 2012 Eighth International Conference on Computational Intelligence and Security, 2012, pp. 568-572.

[10] Buyya R., Beloglazov, A., Abawajy, J. (2010) "Energy-Efficient Management of Data Center Resources for Cloud Computing: **A Vision, Architectural Elements, and Open Challenges**". In International Conference on Parallel and Distributed Processing Techniques and Applications.

[11] Yatendra Sahu, R.K. Pateriya, Rajeev Kumar Gupta:" **Cloud Server Optimization with Load Balancing and Green Computing Techniques Using Dynamic Compare and Balance Algorithm**" 5th International Conference on Computational Intelligence and Communication Networks, 2013.

[12] Ryan Jansen, Paul R. Brenner:" **Energy Efficient Virtual Machine Allocation in the Cloud An Analysis of Cloud Allocation Policies**" 2011 IEEE.

[13] Open Nebula. http://opennebula.org.

[14] Eucalyptus. http://www.eucalyptus.com.

[15] Yi Han, Tansu Alpcan, Jeffrey Chan, and Christopher Leckie:" **Security Games for Virtual Machine Allocation in Cloud Computing**".

[16] Lee Badger, Tim Grance, RobertPatt, and Corner JeffVoas:" **DRAFT Cloud Computing Synopsis and Recommendations**" May 29, 2012.

[17] Cloudsim, http://www.Cloudbus.org/Cloudsim.

[18] Calheiros, R., Ranjan, R., Beloglazov, A., De Rose, C.A.F., and Buyya, R.: "**CloudSim: a Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms**," Software, Practice and Experience, 2011, 41, (1), pp. 23-50