Elysium Journal
Engineering Research & Management

P-ISSN: 2347-4408
E-ISSN: 2347-4734

VIRTUAL MACHINE SECURITY SCHEME AGAINST CO-RESIDENT ATTACK IN CLOUD COMPUTING

Dr.G.Nalinipriya[1], P.J.Varalakshmi[2]

[1]*Professor, IT, Saveetha Engineering College, Anna university, Chennai, Tamil Nadu, India*
E-mail: India.nalinipriya@saveetha.ac.in

[2]*P.G scholar, IT, Saveetha Engineering College, Anna university, Chennai, Tamil Nadu, India*
E-mail: pjvara@gmail.com

Abstract: Cloud computing provide users and enterprises with various capabilities to store and process their data. Cloud security refers to a broad set of policies, technologies and controls deploy to product data. However, customers can face new security risks when they use cloud computing platforms. Previous works mainly attempt to address the problem by eliminating side channels. However, most of these methods are not suitable for immediate deployment due to the required modifications to current cloud platforms. In this paper, we focus on one such threat the co-resident attack, where malicious users build side channels and extract private information from virtual machines co-located on the same server. We choose to solve the problem from a different perspective, by studying how to improve the virtual machine allocation policy, so that it is difficult for attackers to co-locate with their targets. To give security by using Network analyzer tool and cloud-simulation tool. N/A Tool is using to analyze network problem, detect network misuse by internal and external users. Documenting regulatory compliance through logging all perimeter and endpoint traffic. Gather the network statics report. The Cloud-Sim toolkit supports both system and behavior modeling of Cloud system components such as data centers, virtual machines (VMs) and resource provisioning policies.

Keywords: co-resident attack, virtual machine allocation policy, security metrics modeling, N/A tool, Cloud-Sim tool.

1. INTRODUCTION

Cloud computing is one of today's most appealing technology areas due to its cost-efficiency and flexibility. However, despite significant interests, deploying cloud computing in an enterprise infrastructure offers significant security concerns. Cloud computing is defined as a collection of IT resources (servers, databases, and applications) which are available on an on-demand basis, provided by a service company, available through the internet, and provide resource pooling among multiple users. Security is one of the major concerns against cloud com-putting. From the customer's perspective, migrating to the cloud means they are exposed to the additional risks brought about by the other tenants with whom they share the resources−are these neighbors trustworthy, or they may compromise the integrity of others? This paper con-cent rates on one form of this security problem: the co-resident attack (also known as co-location, co-residence, or co-residency attack). Virtualization is the cornerstone of the developing third party compute industry, allowing cloud providers to instantiate multiple virtual machines (VMs) on a single set of physical resources. Customers utilize cloud resources alongside unknown and untrusted parties, creating the co-resident threat – unless perfect isolation is provided by the virtual hypervisor, there exists the possibility for unauthorized access to sensitive customer information through the exploitation of covert side channels.

Commercial third-party clouds allow businesses to avoid over provisioning their own resources and to pay for the precise amount of computing that they require. Virtualization is key to this model. By placing many virtual hosts on a single physical machine, cloud providers are able to profitably leverage economies of scale and statistical multiplexing of computing resources. While many models of cloud computing exist, the Infrastructure-as-a-Service (IaaS) model used by providers such as Amazon's Elastic Compute Cloud (EC2) service offers a set of virtualized hardware configurations for customers. The sharing of a common physical platform amongst multiple virtual hosts, however, introduces new challenges to security, as a customer's virtual machine (VM) may be co-located with unknown and untrusted parties. Placement on a common platform entails the sharing of physical resources, and leaves sensitive data processed in a cloud potentially vulnerable to the actions of malicious co-residents sharing the physical machine.

Virtual machines (VM) area commonly used resource in cloud computing environments. For cloud providers, VMs help increase the utilization rate of the underlying hardware platforms. For cloud customers, it enables on-demand resource scaling, and outsources the maintenance of computing resources. However, apart from all these benefits, it also brings a new security threat. In theory, VMs running on the same physical server (i.e., co-resident VMs) are logically isolated from each other. In practice, nevertheless, malicious users can build various side channels to circumvent the logical isolation, and obtain sensitive information from co-resident VMs, ranging from the coarse-grained, e.g., workloads and web traffic rates, to the fine-grained, e.g., cryptographic keys. For clever attackers, even seemingly innocuous information like workload statistics can be useful. For example, such data can be used to identify when the system is most vulnerable, i.e., the time to launch further attacks, such as Denial-of-Service attacks.

2. CLOUD CO-RESIDENT ATTACK

In compute clouds, the co-resident threat considers a malicious and motivated adversary that is not affiliated with the cloud

P-ISSN: 2347-4408
E-ISSN: 2347-4734

provider. Victims are legitimate cloud customers that are launching Internet facing instances of virtual servers to do work for their business. The adversary, who is perhaps a business competitor, wishes to use the novel abilities granted to him by cloud co-residency to discover valuable information about his target's business. This may include reading private data or compromising a victim machine. It could also include subtler attacks such as performing load measurements on the victim's server or launching a denial of service attack. Masquerading as another legitimate cloud customer, the adversary is free to launch and control an arbitrary number of cloud instances .As is necessary for the general use of any third party cloud, the cloud infrastructure is a trusted component.

The co-resident attack discussed in this paper comprises the following two steps. First, the attacker has a clear set of target VMs, and their goal is to co-locate their VMs with these targets on the same physical servers. Second, after co-residence is achieved, the attacker will construct different types of side channels to obtain sensitive information from the victim. Note that this is different from [5], [6], [7], where attackers do not have specific targets, and their goal is to obtain an unfair share of the cloud platform's capacity. In order to co-locate with the targets, the attacker can either use a brute-force strategy: start as many VMs as possible (the number may be limited by the cost), or take advantage of the sequential and parallel locality in VM placement. It has been shown in [1] that in the Amazon EC2 cloud[8], if one VM is started immediately after another one is terminated, or if two VMs are launched almost at the same time, it is more likely that these two VMs are allocated to the same server.

3. RELATED WORK

There are many works corresponds to this area. Yinqian Zhang et al described about "Home Alone: Co-Residency Detection in the Cloud via Side-Channel Analysis" [1]. Home Alone, a system that lets a tenant verify its VMs' exclusive use of a physical machine. The key idea in Home Alone is to invert the usual application of side channels. Rather than exploiting a side channel as a vector of attack, Home Alone uses a side-channel (in the L2 memory cache) as novel, defensive detection tool. By analyzing cache usage during periods in which "friendly" VMs coordinate to avoid portions of the cache, a tenant using Home Alone can detect the activity of a co-resident "foe" VM. Key technical contributions of Home Alone include classification techniques to analyze cache usage and guest operating system kernel modifications that minimize the performance impact of friendly VMs sidestepping monitored cache portions. Our implementation of Home Alone on Xen-PVM requires no modification of existing hypervisors and any special action or cooperation by a cloud provider.

Adam Bates et al explain about "Detecting Co-Residency with Active Traffic Analysis Techniques" [2]. Co-resident watermarking, a traffic analysis attack that allows a malicious co-resident VM to inject a watermark signature into the network flow of a target instance. This watermark can be used to exfiltrate and broadcast co-residency data from the physical machine, compromising isolation without reliance on internal side channels. As a result, our approach is difficult to defend without costly underutilization of the physical machine. We evaluate co-resident watermarking under a large variety of conditions, system loads and hardware configurations, from a local lab environment to production cloud environments (Future grid and the University of Oregon's ACISS). We demonstrate the ability to initiate a covert channel of 4 bits per second, and we can confirm co residency with a target VM instance in less than 10 seconds. We also show that passive load measurement of the target and subsequent behavior profiling is possible with this attack. Our investigation demonstrates the need for the careful design of hardware to be used in the cloud.

Yi Han yet al described about "Security Games for Virtual Machine Allocation in Cloud Computing" [3]. Concentrates on the co-resident attack, where malicious users aim to co-locate their virtual machines (VMs) with target VMs on the same physical server, and then exploit side channels to extract private information from the victim. Most of the previous work has discussed how to eliminate or mitigate the threat of side channels. However, the presented solutions are impractical for the current commercial cloud platforms. We approach the problem from a dif-ferent perspective, and study how to minimize the attacker's possibility of co-locating their VMs with the targets, while maintaining a satisfactory workload balance and low power consumption for the system. Specifically, we introduce a security game model to compare different VM allocation policies. Our analyze shows that rather than deploying one single policy, the cloud provider de-creases the attacker's possibility of achieving co-location by having a policy pool, where each policy is selected with a certain probability. Our solution does not require any changes to the underlying infrastructure. Hence, it can be easily implemented in existing cloud computing platforms.

Rodrigo N. Calheiros et al explains about "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms" [4]. The recent efforts to design and develop Cloud technologies focus on defining novel methods, policies and mechanisms for efficiently managing Cloud infrastructures. To test these newly developed methods and policies, researchers need tools that allow them to evaluate the hypothesis prior to a real deployment in an environment, where one can reproduce tests. Simulation-based approaches in evaluating Cloud computing systems and application behaviors offer significant benefits, as they allow Cloud developers: (i) to test the performance of their provisioning and service delivery policies in a repeatable and controllable environment free of cost; and (ii) to tune the performance bottlenecks before real-world deployment on commercial Clouds. To meet these requirements, we have developed the CloudSim toolkit for modeling and simulating extensible Clouds. As a

P-ISSN: 2347-4408
E-ISSN: 2347-4734

completely customizable tool, it allows extension and definition of policies in all the components of the software stack, thereby making it a suitable research tool that can handle the complexities arising from simulated environments.

4. PROPOSED WORK

We have proposed a prototype of such a secure policy, called the previous-selected-server-first policy (PSSF). However, this prototype policy only focuses on the problem of security, and hence has obvious limitations in terms of:

4.1. WORKLOAD BALANCE

Workload here refers to the VM requests. From the cloud provider's point of view, spreading VMs among the servers that have already been switched on can help reduce the probability of servers being over-utilized, which may cause SLA (service level agreement) breaches. From the customer's perspective, it is also preferable if their VMs are distributed across the system, rather than being allocated together on the same server. Otherwise, the failure of one server will impact all the VMs of a user.

4.2. POWER CONSUMPTION

It has been estimated that the power consumption of an average data centre is as much as 25,000 households [8], and it is expected to double every 5 years [9]. Therefore, managing the servers in an energy efficient way is crucial for cloud providers in order to reduce the power consumption and hence the overall cost. This has also been the focus of many previous works [10], [11], [12], [13], [14].

In this paper, we take all three aspects of security, workload balance and power consumption into consideration to make PSSF more applicable to existing commercial cloud platforms. Since these three objectives are conflicting to some extent, we improve our earlier policy by applying multi-objective optimization techniques. In addition, we have implemented PSSF on the simulation environment CloudSim[15], [16],as well as on the real cloud platform Open Stack [17], and performed large scale experiments that involve hundreds of servers and thou-sands of VMs, to demonstrate that it meets the requirements of all three criteria.

Specifically, our contributions include: (1) we define secure metrics that measure the safety of a VM allocation policy, in terms of its ability to defend against co-resident attacks; (2) we model these metrics under three basic but commonly used VM allocation policies, and conduct extensive experiments on the widely used simulation plat-form CloudSim[15], [16]to validate the models; (3) we propose a new secure policy, which not only significantly decreases the probability of attackers co-locating with their targets, but also satisfies the constraints in workload balance and power consumption; and (4) we implement and verify the effectiveness of our new policy using the popular open-source cloud software Open Stack[17], as well as on CloudSim.
The rest of the paper is organized as follows. In Section 2,we give a survey of previous work on co-resident at-tacks, and current VM allocation policies. In Section 3, we describe our research aim and formally define the problem. In Sections4 and 5, we model the security metrics under three existing allocation policies, and give an experimental verification .we introduce our new policy and summaries the test results on CloudSim. We present the implementation on Open-Stack .Finally, Section 8concludes the paper, and gives directions for future work.

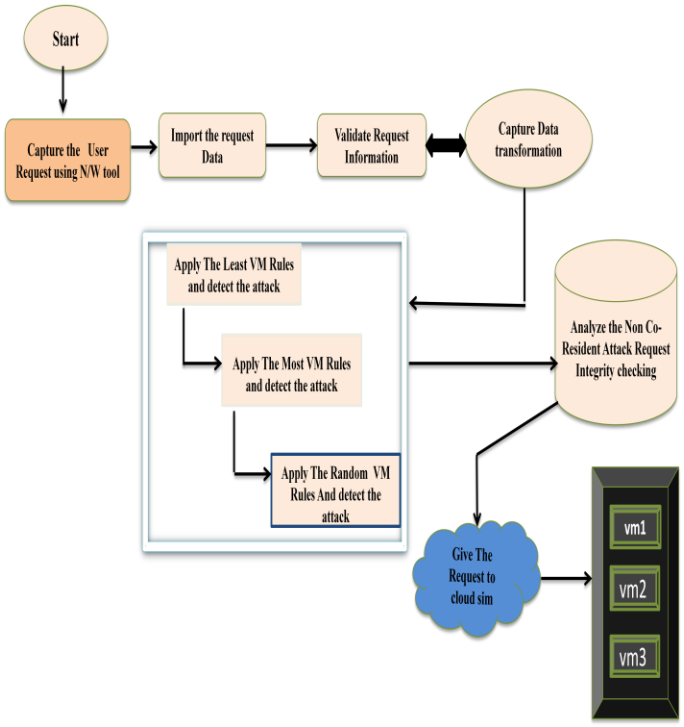

Fig.1 System Architecture Diagram

5. ATTACK SCENARIO

Given this attack scenario, our new policy should satisfy the following objectives:

1. *Security*−Under the new policy, the attacker has to start a large number of VMs to achieve a non-zero efficiency or coverage rate, i.e., *VMmin* is high. In addition, the coverage rate does not increase or increases very slowly. In order to achieve these two points, one extreme case is that VMs of different users are never allocated to the same server. Based on such an idea, we minimize the average number of users per server.

2. *Workload balance*−As we mentioned in the introduction, the importance of workload balance is twofold. For cloud providers, evenly distributing VMs helps decrease the probability of servers being over-utilized. For

P-ISSN: 2347-4408
E-ISSN: 2347-4734

simplicity, in our new policy we use the number of running VMs per server as the criterion to spread the workload (the same as the Least VM policy). In addition, customers would also prefer if their VMs are not all allocated together on the same server. In other words, on average the number of servers that host a user's VMs should be maximized.

3. *Power consumption*−How to effectively reduce the power consumption is a crucial issue for cloud providers[12].Different techniques of energy aware VM placement has been widely discussed in previous papers[9], [10], [11], [12], [13], [14].In order to simplify the problem, here we only consider the most straightforward approach minimizing the number of running servers.

6. A NEW BALANCES VM-ALLOCATION POLICY

A lesson from the previous results is that if the number of servers that each user's VM scan be assigned to is limited, so that the target VMs are less exposed to the attacker, then the impact of co-resident attacks will be mitigated. Based on this idea, we design anew balanced policy, *Previously-selected-servers-first*(PSSF).We introduce step by step how PSSF satisfies the objectives of security, workload balance and power consumption.

Algorithm: Previously selected servers first(PSSF) policy

Step 1: PSSList={},NPSSList={}
Step 2: foreach server s_i in S
Step 3: if (s_i has enough remaining resources)
Step 4: if(s_i already hosts or once hosted u's VMs)
Step 5: if(s_i hosts less than N* of u's VMs)
Step 6: PSSList.add(s_i)
Step 7: else
Step 8: NPSSList.add(s_i)
Step 9: if(!PSSList.isEmpty())
Step 10: returen PSSList.get(random(PSSList.size())) else
Step 11: Sort(NPSSList,group index,resources left)
Step 12: Sort(NPssList,group index,resources left)
Step 13: i=the number of servers with the same group index and remaining resources as the first server in NPSSList(NPSSList.get(0))
Step 14: Mark NPSSList.get(random(i)) as "previously selected" for u,and return it.

1. Security− In order to minimise the average num-ber of users per server, when a user *ui*creates new VMs, they will first be assigned to those servers that already host or once hosted VMs started by*ui*(i.e., previously selected servers).

2. Workload balance− In the following three circum-stances, the new VMs will not be assigned to previously selected servers: (1) every previously selected server already hosts *N**VMs of *ui*, (2) none of the previously selected servers has enough re-sources left, and (3) the user has never started VMs before. In these three cases, PSSF will spread the workload instead, e.g., choose the servers with the least number of VMs.

3. Power consumption–One main reason why the Least VM policy and the Random policy perform poorly in power consumption is that an excessive number of servers are switched on. The most straightforward way to minimize the number of running servers is stacking, or in other words, al-locating new VMs to the same server until there is not enough remaining resources. However, clearly this breaks the rule of workload balance. There-fore, we propose a compromise solution: logically divide all servers into groups of *NG*; within each group, the workload is spread; the next group of servers will not be started until servers in all the Another reason why we choose to spread the work-load, instead of concentrating or randomly distributing it is that it further helps lower the probability of attackers co-locating with the targets. Consider the following example: the victim user starts ten VMs, and they are equally assigned to two servers,*s1*and *s2*. As a result, these two servers host more VMs than the other servers now, and it is unlikely for further VM re-quests (from other users) to be assigned to them until all other servers also host the same number of VMs. Even then, it is difficult for attackers to achieve co-residence, since the target VMs are allocated together, and as a result are less exposed.

7. SIMULATION EXPERIMENT ON CLOUD-SIM

We start by ignoring the constraints on work load balance (*W*) and power consumption (*P*), and only considering the security objective, which is to minimize the average number of users per server. We denote this as "PSSF + Least VM". As can be seen from Fig. 2, 3 and 4, all the values of the three security metrics are quite close to zero, regardless of the number of servers started by the attacker; the reason being that in this case it is like every user has their own dedicated servers. We continue by adding the workload balance constraint, and denote this as "PSSF + Least VM + Limit per user". In our simulation experiment, the limit is set to 3(i.e., every server can only host no more than three VMs from the same user, which we believe is strict enough). We can see from the results that even though this constraint has an obvious negative impact in terms of security, all the metrics are still significantly lower than those under the existing policies .In addition, we calculate for each server the number of times they are selected. The standard deviations are almost the same−in the range of−under this policy as under the Least VM poli-cy, which

P-ISSN: 2347-4408
E-ISSN: 2347-4734

indicates that these two policies perform similarly in balancing the workload.

8. CONCLUSION

This paper provides a new perspective to counter the co-resident attack. Instead of looking for solutions after at-tackers co-locate with their targets, cloud providers can mitigate the threat by minimizing the probability of at-tackers co-locating with the targets. Specifically, we first compare the difficulty of achieving co-residence under three basic yet widely used VM allocation policies, and find that if oversubscription is enabled, and the servers are properly configured, the Most VM policy, or more generally speaking stacking the workload, performs better than the other options1. In addition, we propose and implement a new policy that is effective not only in de-fending against the co-resident attack, but also balancing the workload, and decreasing the power consumption. In the future, we will take into consideration addition-al practical factors to improve the policy: (1) distributed scheduling – we only consider the centralized version of different allocation policies in this paper, and assume that the current system state is always known; (2) live VM migration – live VM migration may enable the attacker to exploit the potential loopholes in the migration algorithm to increase their probability of co-locating with the tar-gets; (3) testing the policy in larger cloud computing sys-tems that consist of multiple data centers.

REFERENCES

[1] Yinqian Zhang, Ari Juels, Alina Oprea (2011) “Home Alone: Co-Residency Detection in the Cloud via Side-Channel Analysis” 2011 IEEE Symposium on Security and Privacy.

[2] Adam Bates, Benjamin Mood, Joe Pletcher, Hannah Pruse, Masoud Valafar (2010) “Detecting Co-Residency with Active Traffic Analysis Techniques”.

[3] Yi Han, Tansu Alpcan, Jeffrey Chan, Christopher Leckie, (2011) “Security Games for Virtual Machine Allocation in Cloud Computing”.

[4] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, C´esar A. F. De Rose (2010) “CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms”

[5] F. F. Zhou, M. Goel, P. Desnoyers, and R. Sundaram, "Scheduler Vulnerabilities and Coordinated Attacks in Cloud Computing," Proc. Tenth IEEE International Symposium on Network Computing and Applications (NCA 2011) pp. 123-130, 2011.

[6] V. Varadarajan, T. Kooburat, B. Farley, T. Ristenpart, and M. Swift, "Resource-Freeing Attacks: Improve Your Cloud Performance (at Your Neighbor’s Expense)," Proc. ACM Conference on Computer and Communications Security (CCS 2012), pp. 281-292, 2012.

[7] Z. Yang, Z. Yang, H. Fang, Y. Wu, C. Li, B. Zhao, and H. H. Huang, "Understanding the Effects of Hypervisor I/O Scheduling for Virtual Machine Performance Interference," Proc. Fourth IEEE International Conference on Cloud Computing Technology and Science (Cloud Com 2012), pp. 34-41, 2012.

[8] "Amazon Elastic Compute Cloud (EC2)," http://aws.amazon.com/ec2/.

[9] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, "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), pp. 199-212, 2009.

[10] P. Graubner, "Energy-efficient Management of Virtual Machines in Eucalyptus," Proc. Fourth IEEE International Conference on Cloud Computing (CLOUD 2011), pp. 243-250, 2011.

[11] R. Jansen, and P. R. Brenner, "Energy Efficient Virtual Machine Allocation in the Coud: An Analysis of Cloud Allocation Policies," Proc. International Green Computing Conference and Workshops (IGCC 2011), pp. 1-8, 2011.

[12] K. Le, R. Bianchini, J. Zhang, Y. Jaluria, J. Meng, and T. D. Nguyen, "Reducing Electricity Cost through Virtual Machine Placement in High Performance Computing Clouds," Proc. International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1-12, 2011.

[13] K. Mills, J. Filliben, and C. Dabrowski, "Comparing VM-Placement Algorithms for On-Demand Clouds," Proc. Third IEEE International Conference on Cloud Computing Technology and Science (Cloud Com 2011), pp. 91-98, 2011.

[14] A. Beloglazov, J. Abawajy, and R. Buyya, “Energy-aware Resource Allocation Heuristics for Efficient Management of Data Centers for Cloud Computing,” *Future Generation Computer Systems*, vol. 28, no. 5, pp. 755-768, 2012.

[15] "CloudSim," http://www.cloudbus.org/cloudsim/.

[16] R. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, “CloudSim: a Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms,” *Software, Practice and Experience*, vol. 41, no. 1, pp. 23-50, 2011.

[17] "Open Stack," http://www.openstack.org/.

P-ISSN: 2347-4408
E-ISSN: 2347-4734