AN IMPROVED RELEVANCE VECTOR MACHINE WITH SPATIAL FUZZY CLUSTERING BASED DEFENSE MECHANISM TO DEFEND AGAINST CO-RESIDENT DOS ATTACKS IN CLOUD COMPUTING

[1] Rethishkumar S,[2] Dr.R.Vijayakumar

[1] Ph. D Scholar,[2] Professor

[1] School of Computer Sciences, Mahatma Gandhi University, Kottayam, Kerala

Abstract : Cloud computing is taking the technology world by storm because of the varieties of services offered by the cloud service providers (CSPs). Despite numerous benefits offered by CSPs, there are some security issues that may dissuade users from using it. In this service, different virtual machines (VMs) share the same physical resources, these VMs are known as co-resident VMs. The shared physical resources pose a significant threat to the users. As resources may belong to competing organizations as well as unknown attackers. From the perspective of a cloud user, there is no guarantee whether the co-resident VMs are trustworthy. The shared resources make privacy and perfect isolation implausible, which paves the way for co-resident attacks, where a VM attacks another co-resident VM. There is a risk that a covert side channel can be used to extract another user's secret information or launch Denial of Service (DoS) attacks. In this paper, an Improved Relevance Vector Machine with Spatial Fuzzy Clustering (IRVM-SFC) based defense mechanism is proposed for minimizing the co-resistance DOS attacks by making it difficult for attackers to initiate attacks. In this process, first, the attacker behavior is analyzed by using Previously Selected Server First (PSSF) VM allocation strategy. Then, the partial labeling is done by using SFC scheme to partially distinguish the users as legal or malicious. After that, an IRVM scheme is proposed for classifying all users into three categories like high risk (i.e. malicious), medium risk (i.e. uncertain), and low risk. In IRVM, the Mosquito Flying behaviour based swarm intelligence Optimization (MFO) approach used to optimize the kernel functions of parameters to improve the training process. Finally, aStackelberg Game Approachis presented to increase the cost of launching new VMs thus minimizing the probability of initiating co-resident DOS attack. The experimental results show that the proposed IRVM-SFC attained high performance results compared than existing VM allocation schemes.

IndexTerms- Cloud security, Defence mechanism, Improved Relevance Vector Machine, Spatial Fuzzy Clustering, PSSF**..**

I. Introduction

Virtual machines (VM) are one of the most crucial and fundamental components to cloud computing systems. For cloud providers, VMs facilitate the efficient utilisation of the hardware platforms. For cloud customers, VMs reduce the maintenance overhead of computing resources.

However, similar to traditional web servers, VMs can also be exposed to various types of security threats. Some forms of attacks, such as brute force SSH attacks, are relatively easy to block, because of the evidence left from failed attempts, which can be found in authorisation logs. In contrast, other forms of attacks are much harder todetect due to their stealthy nature, which leaves little trace in the system logs. In this paper, we focus on one such stealthy security threat: the co-resident attack (also known as co-residence, co-residency, or co-location attack).

Virtualisation techniques [1] provide logical isolation between VMs that locate on the same server (i.e., coresident VMs). This means that programs running on one VM should not interfere with other programs that run on co-resident VMs. Nevertheless, this can happen in real cloud systems. For example, the execution time of cache read operations is to a large extent influenced by the cache utilisation rate [2]. As a result, malicious users can build various types of side channels [2-7] between their VM and the target VM on the same server, and then extract sensitive information from the victim. This is what we call a co-resident attack.
Although it is not easy to perform this type of attack in real cloud environments [8], the potential security risk is still very high, as it has been demonstrated in a number of papers [2-7] that various types of private information including cryptographic keys can be leaked to malicious users. For clever attackers, even seemingly innocuous information like workload statistics [3] can be useful. For example, they can identify when the system is most vulnerable based on the workload data, and launch further attacks, such as Denial-of-Service attacks.

In order to defend against co-resident attacks, most previous work [9-17] has concentrated on preventing the construction of side channels. However, the main problem with these methods is that they often require substantial changes to be made to existing systems which can be costly for current cloud providers. For example, Vattikonda et al. [13] and Wu et al. [14] propose to remove or modify the high resolution clocks that many side channels rely on, while Jin et al. [10] and Szefer et al. [12] redesign the architecture of cloud computing systems. More recently, Zhang et al. [16] propose to perform periodic time-shared cache cleansing, in order to make the side channel noisy. In addition, Varadarajan et al. [17] show that a scheduling mechanism called minimum run time (MRT) guarantee is effective in preventing cache-based side channel attacks. These two methods require fewer changes to existing cloud platforms and hence are easier to deploy.

In addition to the above methods, Sundareswaran and Squcciarini [18] and Yu, et al. [6] proposed to identify abnormalities in CPU and RAM utilisation, system calls, and cache miss activity, when malicious users adopt the Prime+Probe technique [19, 20] to obtain information from the victim. Accordingly, they design different defence mechanisms to detect these features and mitigate the attack.

We look at the problem of co-resident attacks from a new perspective. Instead of studying the possible countermeasures after attackers co-locate with their targets, we intend to find an effective way to prevent attackers from achieving co-residence, as it has been shown that attackers can obtain a relatively high success rate of co-location in existing commercial cloud platforms [2, 21]. Specifically, we focus on the design of the VM allocation policy (also known as VM placement), since from a cloud provider's point of view, the allocation policy is the most direct way they can influence the probability of colocation.

To the best of our knowledge, only [22] adopts a similar method. In their algorithm, all servers are labelled aseither open or closed, where open (closed) means the server can (cannot) receive more VMs. At any time, the algorithm keeps a fixed number (N_{open}) of servers open, and randomly allocates a new VM to one of these servers. If the selected server cannot take more VMs due to this allocation, it will be marked closed, and a new server will be opened.

Compared with their method, our work is more targeted at the attacker. We have proposed three different solutions, in the order of the required modifications (it should be noted that all our work, including this paper, only discusses the defence method within a single datacentre): First, we search for the most secure option among the existing allocation policies [23, 24]. Second, in order to make it more difficult for the attacker to infer the defender's allocation policy and optimise their strategy accordingly, we introduce a game theoretical approach to combine multiple policies to increase the unpredictability in VM allocation [23]. Third, we propose and implement a secure policy named PSSF (Previously Selected Server First). The key idea is to give a higher priority to servers that already host or once hosted VMs from the same user, when a new VM request is being processed. We show that PSSF significantly decreases the probability of attackers co-locating with their targets, while satisfying constraints on workload balance and power consumption [24, 25].

In contrast, we make the following contributions in this paper:

1. We take into consideration the differences between the behaviours of attackers and legal users. By applying clustering analysis, and constructing IRVMs, we classify all users into three categories − high risk (i.e. malicious), medium risk (i.e. uncertain), and low risk (i.e. legal) − and modify the VM allocation process accordingly.

2. We model the problem as a Stackelberg Game Approach[26], with the aims of maximising the overall cost for the attacker, and minimising their probability of achieving co-residence.

3. We perform an accurate quantitative analysis of the attacker's cost in different situations, which is important in finding the equilibrium strategies for both players.

4. We show that under this defence mechanism, attackers are forced to behave normally as other users. As a result, their overall cost is increased by one to two orders of magnitude, according to our extensive experiments on CloudSim [27, 28].

5. We demonstrate that instead of fixing their strategy, the defender should use an adaptive strategy in order to maximise their utility.

The overall organization of the research work is given as follows: The remainder of the paper is organized as follows: section 2 describes various past researches related to this research work. Section 3 explains the proposed defense mechanism. Section 4 presents the evaluation results of the proposed model while the section 5 makes a conclusion about this research work.

I. RELATED WORKS

In this section, some existing security schemes against co-resident DOS attack in cloud computing has been discussed and their merits and demerits also presented.Hanet al., [23] approached the problem from a different perspective, and study how to minimise the attacker's possibility of colocating 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 analysis shows that rather than deploying one single policy, the cloud provider decreases 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.

Khorshed et al., [29] tried to classify DoS attacks in cloud computing using different rule-based learning. They claim to identify attacks and propose a remedy from the client's perspective. The principal idea behind their experiments is to check the CPU performance andnetwork usages of the physical machine. The strategy involves running a statistical analysis among a few classification Machine Learning algorithms already implemented in Weka, to observe which algorithm performs the best in detecting the DoS attacks. However, a cloud user is usually unaware of the physical machine on which his/her application is running and it is not unclear how the proposed remedy removed the ongoing attacks after the detection.

Bedi& Shiva [30]focused on one such specific kind of attack, namely a denial of service (DoS), where an attacker congests a bottleneck network channel shared among virtual machines (VMs) coresident on the same physical node in the cloud infrastructure. We evaluate the behavior of this shared network channel using Click modular router on DETER testbed. We illustrate that game theoretic concepts can be used to model this attack as a two-player game and recommend strategies for defending against such attacks. Huan Liu et al., [31] proposed a DoS prevention mechanism for clouds considering a possible new form of DoS attack. Usually, attackers target the uplink capacity as it is smaller than the aggregated capacity, and send spoof packets to other hosts in a remote router.

Hanet al., [32] focused on one risk at the virtual machine level, the co-resident attack, where by constructing various types of side channels, malicious users can obtain sensitive information from other virtual machines that co-locate on the same physical server. Most previous work has focused on the elimination of side channels, or more generally speaking, the possible countermeasures after attackers co-locate with their targets. In contrast, we provide a different perspective, and propose a defence mechanism that makes it difficult and expensive for attackers to achieve co-residence in the first place. Specifically, we (1) identify the potential differences between the behaviours of attackers and legal users; (2) apply clustering analysis, and semi-supervised learning techniques to classify users; (3) model the problem as a two-player security game, and give a detailed analysis of the optimum strategies for both players; (4) demonstrate that the attacker's overall cost is increased dramatically by one to two orders of magnitude as a result of our defence mechanism.

Qiuet al., [33] proposed a co-residency-resistant VM deployment strategy and define four thresholds to adjust the strategy for security and load balancing. Moreover, two metrics(VM co-residency probability and user co-residency coverage probability) are introduced to evaluate the deployment strategy. Finally, we implement the strategy and run experiments on both OpenStack and CloudSim. The results show that our strategy can reduce VM co-residency by 50% to 66.7% and user co-residency by 50% to 66% compared with the existing strategies.

Atyaet al., [34] first undertake an experimental study on Amazon EC2 to obtain an in-depth understanding of the sidechannels an attacker can use to ascertain co-residency with a victim. Here, we identify a new set of stealthy side-channel attacks which, we show to be more effective than currently available attacks towards verifying co-residency. Based on the study, we develop a set of guidelines to determine under what conditions victim VM migrations should be triggered given performance costs in terms of bandwidth and downtime, that a user is willing to bear. Via extensive experiments on our private in-house cloud, we show that migrations using our guidelines can limit the fraction of the time that an attacker VM co-resides with a victim VM to about 1 % of the time with bandwidth costs of a few MB and downtimes of a few seconds, per day per VM migrated.

Yu et al., [35] investigated the isolation enhancement scheme from the aspect of *virtual machine* (VM) management. The security-awareness VMs management scheme (SVMS), a VMs isolation enhancement scheme to defend against side channel attacks, is proposed. First, we use the *aggressive conflict of interest relation* (ACIR) and *aggressive in ally with relation* (AIAR) to describe user constraint relations. Second, based on the Chinese wall policy, we put forward four isolation rules. Third, the VMs placement and migration algorithms are designed to enforce VMs isolation between the conflict users. Finally, based on the normal distribution, we conduct a series of experiments to evaluate SVMS. The experimental results show that SVMS is efficient in guaranteeing isolation between VMs owned by conflict users, while the resource utilization rate decreases but not by much.

II. PROPOSED METHODOLOGY

In this section, the proposed IRVM-SFC based defense mechanism has been discussed. The step by step process of proposed scheme has been explained in given below sub section.

a. Problem Definition

We consider the following attack scenario: in a cloud computing system, attacker *A*'s targets are the VMs started by legal user *U*, and *A*intends to minimise thecost ofco-locatingtheir VMs with as many VMs of *U*as possible. The defender *D*knows the existence of the attack, but cannot identify who the attacker or victim is. Instead, *D*deploys the following defence mechanisms shown in Figure.1.

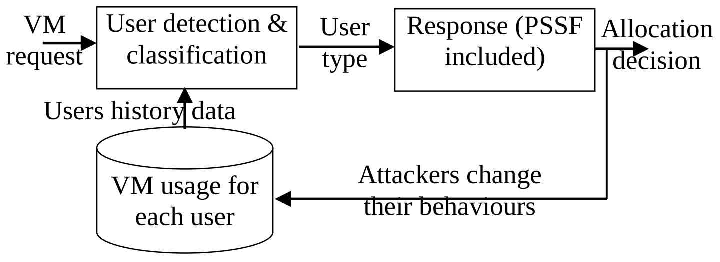

Figure 1: Architecture of proposed IRVM-SFC based co-resident defense scheme

The defence mechanism consists of two main parts: the detection module and the response module. When a user requests to start a new VM, the detection module loads historical data from the database, classifies the user into one of the three types like low or medium or high risk, and sends the result to the response module. The latter limits the available servers according to the result, so that VMs of any user only co-locate with VMs started by users of the same type. In addition, the response module sets the parameter in the VM allocation policy PSSF, which finally selects a server within the limited set.

It is likely that attackers will adapt the way they start VMs according to the allocation decisions. Meanwhile, the defence system will also update the database of VM usage regularly, which in turn may change the allocation decision. Therefore, in order to analyse how to optimize this adversarial learning problem, we model the problem as a two-player securitygame (*G*) between the attacker (*A*) and the defender(*D*).We define this game theoretic model in terms of the action set and utility function of the attacker and defender.

b. Action set of the attacker

Duringanattack, the attacker *A*needs to decide:(1) the number of VMs they start (*N*), (2) when they start these VMs (*T*), and (3)the length of time for whicheach VM runs (*L*). Therefore, the action set of the attacker is $AS^A = \{< N, T, LEN >\}$, where $T = (t_1, t_2, \dots, t_N)$ and $L = (l_1, l_2, \dots, l_N)$.

Action set of the defender

The main problem for the defender is to classify all the accounts. In general, the dataset of VM usage that we have is unlabelled. Consequently, the classification process consists of the following steps:

1. Clustering. Rather than labelling each individual account, it would be easier to label groups of ac-counts, which are much smaller in number. Hence, we first cluster the accounts into related groups. Weare interested in the marked regions, and an intuitive observation from the figure is that these two regions are relatively dense. Hence, we choose the density based algorithm SFC to cluster the nodes/accounts.

2. Partial labelling. After obtaining a list of clusters, we mark them as low orhigh risk if they exactly match with the characteristics of normal users or attackers. In other words, we use the clustering algorithm to help us label those nodes that are highly likely to be legal or malicious.

3. IRVM learning. Once we have a partially labelled dataset, we use the RVM with MFO to classify all the nodes.

In addition, the defender also needs to set the parameter in the VM allocation policy. Recall that in our PSSF policy, one constraint requires that no more than a certain number of VMs (*N**) from one single user are hosted on the same server.

Therefore, the action set of the defender is $AS^D = \{< P_C, P_S, N^* >\}$, where: (1) $P_C$and $P_S$are the parameter vectors for the algorithms of SFC and IRVM; (2) $N^* = (N_1^*, N_2^*, N_3^*)$, where $N_i^*(1 \leq i \leq 3)$ applies to users of low, medium, or high risk, respectively.

Utilities of the attacker and the defender
The attacker aims to maximise the number of VMs that co-locate with the targets, with the minimum cost possible.Before we define the attacker's and defender's utility functions, we first define the following functions:

1. *TP*(*N, T, LEN, PC, PS*): the type into which the attacker's account is classified, when it starts *N*VMs during one attack, each of which starts at *t*iand runs for *li*(1 ≤ *i*≤ *N*) seconds, while the defender sets theirparameters to be *PC*and *PS*. The range of *TP*(·) is {"high risk", "medium risk", "low risk"};

2. *F*(*N, N**(*TP*(·))):the coverage rate, i.e., the percent-ageof target VMs that are co-located by the attacker *A*, when *A*starts *N*VMs, and*A*is classified intothetype *TP*(·), while the parameter *N**in the VM allocation policy is set to *N**(*TP*(·)). As we mentioned earlier, the range of *N**(*TP*(·)) is $\{N_1^*, N_2^*, N_3^*\}$;

3. *C*(*N, L*): the total cost for the attacker *A*to start*N*VMs,each of which runs for *li*(1 ≤ *i*≤ *N*) seconds. Note that the attacker *A*must belong tothe same type as their target account, and this requirement influences the length of time for which each of *A*'s VMs runs.

Given these functions, the attacker's utility is defined as

$$u^A = w_A, F\left(N, N^*(TP(.))\right) - (1 - w_A).C(N, L) \qquad (1)$$

Where $0 \leq w_A \leq 1$is aweightthat reflects the tradeoff be-tween the probabilityof co-locationand the cost of the attack.

From the attacker action set, it can be found that the attacker is capable of triggering their targets to launch new VMs, capable of compromising a low risk account, and only starts same type of VMs. Hence based on these attacker behaviors, the defense process is defined. The defense process begins with determination of attacker behavior. Then the user accounts are clustered using SFC and then labeled partially as low, medium or high risks. Finally the accounts are classified using IRVM-MFO. Based on the classified results, the cost for the initialization of new VMs through new accounts is maximized making it difficult for the attacker to co-locate the target VMs.

Spatial Fuzzy C-Means Clustering

Clustering is used to classify key frames into identical groups in the process of action recognition. It also exploits segmentation which is used for quick bird view for any kind of problem. K-Means is a well-known partitioning method. Useful key frames are classified as belonging to one of k groups, k chosen a priori. Cluster membership is determined by calculating the centroid for each group and assigning each key frame to the group with the closest centroid. This approach minimizes the overall within-cluster dispersion by iterative reallocation of cluster members. Based on this, human action recognition dataset is implemented. This produces the proven results for video datasets using K-Means clustering. But in same datasets, if different structures exist, it has often found to fail. In general, the fuzzy c-means algorithm is assigned the pixels to fuzzy clusters without any label. Hard clustering methods are used to group pixels to belong exclusively one cluster. But, FCM allows a pixel in more than one cluster depends on the degrees of membership. Summation of membership of each data points in the given datasets should be equal to each other. Let $X = \{x_1, x_2, x_3 \dots, x_n\}$ be the set of data points and $C = \{c_1, c_2, c_3 \dots, c_n\}$ be the set of centers. The following equations 2and 3explain the membership and cluster center updation for each iteration.

$$\mu_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{d_{ij}}{d_{ik}}\right)^{\left(\frac{2}{m}-1\right)}} \qquad 3)$$

$$c_j = \sum_{i=1}^{n}\left(\frac{\left(\left(\mu_{ij}\right)^m x_i\right)}{\left(\mu_{ij}\right)^m}\right) \qquad 4)$$

Where d_{ij} represents the distance between ith data and jth cluster center. c represents the number of cluster m is the fuzziness index μ_{ij} represents the membership of ith data to jth cluster center, n is the number of data points. c_j represents the jth cluster center. Based on this process, the users are cluster as one group.

a. Partial Labeling

Once the list of clusters is obtained, the next step is to compare them with the attacker's potential behaviors, and mark those clusters that are highly likely to be malicious or legal. After the labeling process is completed, the semi-supervised learning method is employed for user account classification.

Improved Relevance Vector Machine (IRVM)

The last step of the classification is to apply semi-supervised learning techniques on the partially labeled dataset obtained from the previous step. RVM model is working based on the Support Vector Machine (SVM) model, but RVM is more robust, training time is less compared than SVM and it contains minimum basis function and is more generalized. RVM performs good classification on the vegetation that contains practically large image samples. The given training inputs$\{IS_i, t_i\}_{i=1}^{n}$, where $IS_i \in R^n, t_i \in \{0,1\}$ and n is the number of samples, IS defined image samples and t is training time. Based on the SVM-like function, the RVM makes predictions for new inputs $\widehat{IS}$ and it attains the form of a linear mixture of basic functions forward via a logistic sigmoid function

$$q(\widehat{IS}, w) = \sigma\left(\sum_{i=1}^{n} \omega_i k(IS_i, \widehat{IS})\right) = \sigma(w^T K) \qquad (4)$$

Where $k(\widehat{IS})$defined as the kernel function vectorand denoted as $k(\widehat{IS}) = \left[k(IS_1, \widehat{IS}) \dots k(IS_n, \widehat{IS})\right]^T$, w indicated the weight vector and is defined as $w = (\omega_1 \dots \omega_n)^T$, and $\sigma(.)$ is the logistic sigmoid function defined as follows:

$$\sigma(a) = \frac{1}{1 + \exp(-a)} \qquad (5)$$

The logistic sigmoid function is defined

$$\sigma(-a) = 1 - \sigma(a) \qquad (6)$$

As a result, the posterior probability can be used in IRVM. The class $c_1$posterior probability can be calculated for the input$\widehat{IS}$ is given below

$$Pr(t = 1|\widehat{IS}) = q(\widehat{IS}, w) \qquad (7)$$

Similarly, the class $c_2$posterior probability can be calculated for the input $\widehat{IS}$ is given below

$$Pr\left(t = 0\middle|\widehat{IS}\right) = 1 - IS(\widehat{IS}, w) \qquad (8)$$

In this process, RVM model can be treated as the posterior probability due to Bayesian probabilistic framework. RVM can be used for Automatic Relevance Determination (ARD) prior over the weight vector w and is defined as a separate hyper parameter α_i for each of the weight parametersω_i.Through the deduction process, a lot of hyper parameters are decided to high values, so that equivalent weights are efficiently forced to zero. Thus the corresponding kernel functions can be pruned out, resulting in a sparse model. The remaining nonzero weights of the inputs IS_i are known as relevance vectors. For an input vector $\widehat{IS}$, the RVM decision model, as defined in Equation(4), based on the $w_{MP}$and RVM can be rewritten as follows

$$q(\widehat{IS}, w_{MP}) = \sigma\left(\sum_{IS_i \in RVM} \omega_i k\left(IS_i\,, \widehat{IS}\right) + \omega_0\right) \qquad (9)$$

As can be seen in Equations (8) and (9), kernel function plays an important role in the RVM decision model. There are several common kernel functions for selection, such as linear, polynomial, sigmoid, Gaussian Radial Basis Function (RBF) and so on. In this improved RVM the Elliptical Radial Basis Function (ERBF) used for kernel function.

$$(p, z) = exp(-\sum_{i=1}^{D}(IS_i - z_i)^2/(\sigma_i^2 . r^2)) \qquad (10)$$

Where pand z are D-dimension feature vectors (i.e. $= (p_1, \dots, p_D)^T, z = (z_1, \dots, z_D)^T$), r is scale factor, σ_i^2 variance. The kernel parameters are optimized using MFO to improve the classification accuracy and reduced the processing time.

Mosquito Flying behaviour based Swarms intelligence Optimization Algorithm (MFO)

Background of MFO: Ruiz-Vanoye and Díaz-Parra [36] introduced the Mosquito Swarm Algorithm (MSA) and it is defined as a meta-heuristics algorithm based on the social behavior of mosquito swarm [36]. Mosquitoes (gnats) have sensors designed to track their prey: A) Chemical sensors, mosquitoes can sense carbon dioxide and lactic acid up to 36 meters away. Mammals and birds give off these gases as part of their normal breathing. Certain chemicals in sweat also seem to attract mosquitoes.

B) Heat sensors, Mosquitoes can detect heat, so they can find warm-blooded mammals and birds very easily once they get close enough. A mosquito swarm exists close to areas with standing water.

C) They go to their pray through flying motion and slide around it to find the most favourable point. This combined effort of flying and sliding motion of the mosquitoes has been imitated in the present algorithm.

Proposed Mosquito Swarm Algorithm

This algorithm has been used to find cluster centre among the high and low accuracy prediction values. The main aid of this algorithm is to cluster the high and low accuracy values and choose the high accuracy value for kernel function. The step by step process of this algorithm is given below

Input: n-number of mosquitoes (i.e kernel parameters)

1. A Mosquito Population Initialized with Chemical Sensors (CS) and Heat Sensors (HS). // here the CS defined the fuzzification parameter and HS defined the number of iterations
2. The initial locations (x) of the mosquitoes (n) generated.
3. The temperature (t) and Maximum Temperature (T) are initialized. // the temperature has been considered as maximum accuracy
4. Find the fitness function based on the maximum temperature
5. Applying the sliding motion to find fitness value
6. Repeated the mosquitoes by parallel or distributed processing
7. Maximum temperature repeat
8. New solutions Generated by adjusting the HS and updating the locations (x).
9. Verified and assigned the feasibility of the solution by the CS.
10. Applying the flying motion
11. Evaluated the new particles
12. Modified the fitness function if new fitness is better
13. The best solution (S) is Selected.
14. While t < T // Maximum Temperature
15. While (n total of mosquitoes)
16. The best solutions are displayed. Based on the above description of mosquito swarm process, the proposed kernel parameters are optimized. It improved the classification accuracy.

Thus the optimal solutions for the IRVM classification are obtained and hence the user accounts are classified accurately.

c. Stackelberg Game Approach

As stated [26], once the user accounts are classified, the game approach is employed for enhancing the defense mechanism. A generic Stackelberg game has two players, a leader and a follower. The leader is also called the defender,whereas the follower is also called an attacker. The leader first commits to a mixed strategy that can be observedby the other agent (the follower or adversary) before the adversary best responds to the leader's mixed strategy. Thedifferent strategies of the optimal mixed strategy are given appropriate weights to arrive at the randomised schedule.The defender randomises the schedules as the attacker can conduct VMs and attack any deterministic strategy.The attacker's behavior, as found earlier, when the attacker starts their first VM (there is no incentive for attackers to start more than one VM at first, as none of them will co-locate with the targets), they are labeled as medium risk. In order to be reclassified as low risk, the attacker has to keep the first VM running before starting more VMs. This is called the initial cost. After being labeled as low risk, the attacker can create as many VMs as they want. However, they have to carefully control the pace, so that they will not be reclassified as medium or even high risk. When it becomes more expensive to keep the current account being considered as low risk than to create a new account (i.e., pay the initial cost again), the attacker will discard the current account.

III. RESULTS AND DISCUSSION

The performance of the proposedIRVM-SFC based scheme is evaluated in CloudSim. The results are compared with theState Observation based Co-Resident DOS Attack Detection (SO-CRDOS-AD), co-location resistant (CLR) algorithm proposed in [37], PSSF based game theoretical approach (referred as PSSF in graphs) [32] and out previous work two-player game approach based defense mechanism (referred as Two-player approach in graphs) in terms of account classification accuracy, precision, recall and attackers overall cost.

Classification accuracy

Accuracy is the percentage corresponding to the correctly done classification of user accounts as legal and malicious users.

$$\text{Accuracy} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{FN} + \text{FP} + \text{TN})} * 100$$

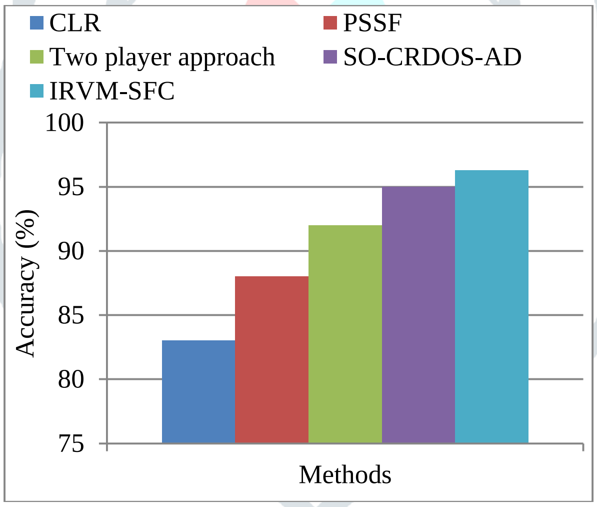

Figure.2. Accuracy comparison

Figure 2 shows the comparison of the defense mechanisms in terms of accuracy. From the graph, it can be found that the proposed IRVM-SFC provides highly accurate classification. The proposed research method IRVM-SFCshows 1.3% increased accuracy than SO-CRDOS-AD, 4.5 % increased accuracy than the two player approach, 9.2 % increased accuracy than PSSF,approach and 15.3% increased accuracy than the CLR approach.

Precision

Precision is the correctness of the classification of the user accounts.

$$\text{Precision} = \frac{\text{TN}}{(\text{TP}+\text{FP})} * 100$$

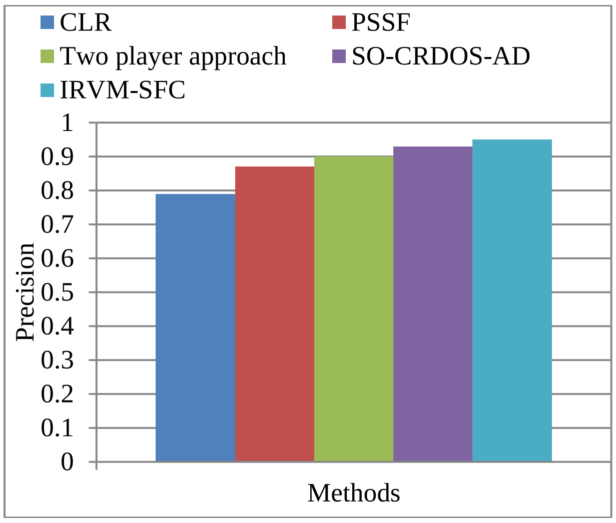

Figure.3. Precision comparison

Figure 3 shows the comparison of the defense mechanisms in terms of precision. From the graph, it can be found that the proposed IRVM-SFCprovides précised classification thanks to the optimal solutions for the large size data center. It is evident that the proposed defense mechanism can avoid the possibility of attacks with greater probability. The proposed research method IRVM-SFCseems to provide 2% increased than SO-CRDOS-AD approach, 5% increased precision rate than two player approach, 8 % improved precision outcome than the PSSF method, and 16 % improved precision outcome than the CLR method.

Recall
Recall is the completeness of the classification done in the cloud user accounts.

$$\text{Recall} = \frac{\text{TN}}{(\text{TP} + \text{FN})} * 100$$

Figure 4 shows the comparison of the defense mechanisms in terms of recall. From the graph, it can be found that the proposed IRVM-SFC provides classification with high recall. The proposed approach increases the defense against co-resident DOS attacks with higher accuracy. Recall of the proposed research method is improved 1.5% increased than SO-CRDOS-AD approach, 4.5% better than the two player approach, 12.5% better than PSSF method and 16.5% better than the CLR method.

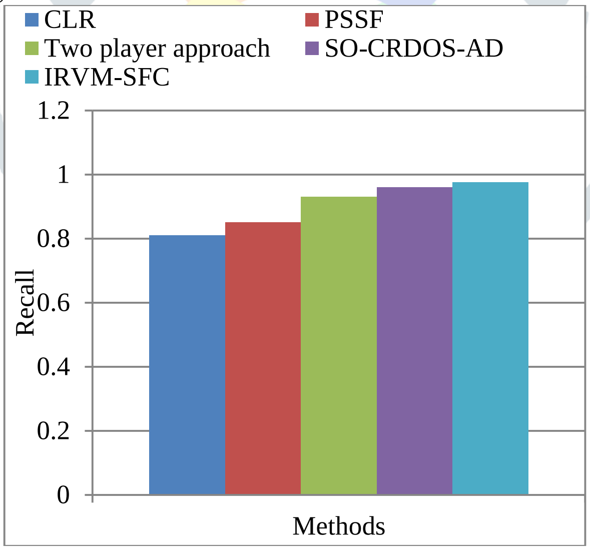

Figure.4. Recall comparison

Attacker's overall cost

This parameter helps in evaluating the cost incurred for initiating an attack i.e. creating a new account for initiating a new VM. The cost is represented in US dollars ($) for common cost evaluation.

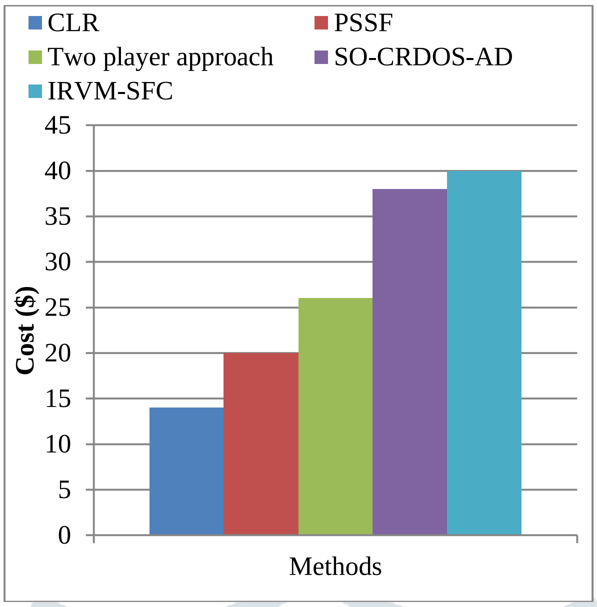

Figure.5. Attacker's overall cost comparison

Figure 5 shows the comparison of the attacker's overall cost for initiating a new VM for beginning a co-resident DOS attack in the presence of evaluated defense mechanisms. From the graph, the cost incurred by the attacker to initiate a co-resident DOS attack is higher in proposed IRVM-SFC than the other methods. It is evident that the proposed defense mechanism makes it difficult for an attacker by making an attack process highly expensive. Thus it forces the attacker to reduce risks and behave as a normal user. The proposed research method shows 2% improved performance thanSO-CRDOS-ADapproach, 14% increased than two player approach, 20% improved performance than the PSSF method and 26% improved performance than the CLR method.

IV. CONCLUSION

In this paper, an IRVM-SFC based defence mechanism is proposed against the co-resident attack in cloud computing environments. The mechanism identifies the behavioural differences between attackers and normal users, and classifies all users into three categories like low, medium and high risk − by applying SFC clustering and IRVM learning techniques. In this way, attackers are forced to behave similarly to legal users (their targets), as we require that only VMs belonging to the same type of users can co-locate with each other. As a result, the attacker's overall cost is increased dramatically by one to two orders of magnitude. While an attacker with unlimited resources may be able to achieve co-residence, in practice there is a cost of launching VMs. We have shown that our approach can increase the attacker's overall cost by one to two orders of magnitude. This makes a significant contribution to the ultimate objective of preventing co-residence attacks in practice. In addition, the defence mechanism builds on our earlier work − a secure VM allocation policy PSSF. The integration of PSSF makes sure that it is difficult for malicious users to co-locate with their targets. In the future, we will further consider the impact of the defence mechanism on normal users, in order to make it as practical as possible.

V. REFERENCE

[1] P. Barham, B. Dragovic, K. Fraser, S. Hand, T.Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the Art of Virtualization," *Operating Systems Review*,vol. 37, no. 5, pp. 164-177, 2003.

[2] 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.

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

[4] Y. Zhang, A. Juels, A. Oprea, and M. K. Reiter, "HomeAlone: Co-residency Detection in the Cloud via Side-Channel Analysis," *Proc.IEEE Symposium on Security and Privacy (SP 2011)*, pp. 313-328, 2011.

[5] Y. Zhang, A. Juels, M. Reiter, and T. Ristenpart, "Cross-VM Side Channels and Their Use to Extract Private Keys," *Proc.19th ACM Conference on Computer and Communications Security (CCS 2012)*, pp. 305-316, 2012.

[6] S. Yu, X. Gui, and J. Lin, "An Approach with Two-stage Mode to Detect Cache-based Side Channel Attacks," *Proc.International Conference on Information Networking (ICOIN 2013)*, pp. 186-191, 2013.

[7] S. Yu, X. Gui, J. Lin, X. Zhang, and J. Wang, "Detecting VMs Co-residency in Cloud: Using Cache-based Side Channel Attacks," *Elektronika ir Elektrotechnika*,vol. 19, no. 5, pp. 73-78, 2013.

[8] M. Green, "The Threat in the Cloud," *IEEE Security & Privacy*,vol. 11, no. 1, pp. 86-89, 2013.

[9] A. Aviram, S. Hu, B. Ford, and R. Gummadi, "Determinating Timing Channels in Compute Clouds," *Proc.ACM Workshopon Cloud Computing Security Workshop (CCSW 2010)*, pp. 103-108, 2010.

[10] S. Jin, J. Ahn, S. Cha, and J. Huh, "Architectural Support for Secure Virtualization under a Vulnerable Hypervisor," *Proc.44th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 2011)*, pp. 272-283, 2011.

[11] J. Shi, X. Song, H. Chen, and B. Zang, "Limiting Cache-based Side-channel in Multi-tenant Cloud using Dynamic Page Coloring," *Proc.41st Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W 2011)*, pp. 194-199, 2011.

[12] J. Szefer, E. Keller, R. Lee, and J. Rexford, "Eliminating the Hypervisor Attack Surface for a More Secure Cloud," *Proc.18th ACM Conference on Computer and Communications Security (CCS 2011)*, pp. 401-412, 2011.

[13] B. Vattikonda, S. Das, and H. Shacham, "Eliminating Fine Grained Timers in Xen," *Proc.Third ACM Workshop on Cloud Computing Security Workshop (CCSW 2011)*, pp. 41-46, 2011.

[14] J. Wu, L. Ding, Y. Lin, N. Min Allah, and Y. Wang, "XenPump: A New Method to Mitigate Timing Channel in Cloud Computing," *Proc.Fifth IEEE International Conference on Cloud Computing (CLOUD 2012)*, pp. 678-685, 2012.

[15] T. Kim, M. Peinado, and G. Mainar-Ruiz, "STEALTHMEM: System-Level Protection Against Cache-Based Side Channel Attacks in the Cloud," *Proc.21st USENIX Security Symposium*, pp. 189-204, 2012.

[16] Y. Zhang, and M. K. Reiter, "Düppel: Retrofitting Commodity Operating Systems to Mitigate Cache Side Channels in the Cloud," *Proc.2013 ACM SIGSAC Conference on Computer & Communications Security (CCS 2013)*, pp. 827-838, 2013.

[17] V. Varadarajan, T. Ristenpart, and M. Swift, "Scheduler-based Defenses against Cross-VM Side-channels," *Proc.23rd USENIX Security Symposium*, pp. 687-702, 2014.

[18] S. Sundareswaran, and A. Squcciarini, "Detecting Malicious Co-resident Virtual Machines Indulging in Load-Based Attacks," *Information and Communications Security*, Lecture Notes in Computer Science Series, S. Qing, J. Zhou and D. Liu, eds., pp. 113-124: Springer International Publishing, 2013.

[19] D. A. Osvik, A. Shamir, and E. Tromer, "Cache Attacks and Countermeasures: The case of AES," *Proc.the Cryptographers' Track at the RSA Conference on Topics in Cryptology*, pp. 1-20, 2006.

[20] E. Tromer, D. A. Osvik, and A. Shamir, "Efficient Cache Attacks on AES, and Countermeasures," *Journal of Cryptology*,vol. 23, no. 1, pp. 37-71, 2010.

[21] V. Varadarajan, Y. Zhang, T. Ristenpart, and M. Swift, "A Placement Vulnerability Study in Multi-Tenant Public Clouds," *Proc.24th USENIX Security Symposium*, pp. 913-928, 2015.

[22] Y. Azar, S. Kamara, I. Menache, M. Raykova, and B. Shepard, "Co-Location-Resistant Clouds," *Proc.Sixth ACM Workshop on Cloud Computing Security*, pp. 9-20, 2014.

[23] Y. Han, T. Alpcan, J. Chan, and C. Leckie, "Security Games for Virtual Machine Allocation in Cloud Computing," *Decision and Game Theory for Security*, Lecture Notes in Computer Science Series, pp. 99-118: Springer International Publishing, 2013.

[24] Y. Han, J. Chan, T. Alpcan, and C. Leckie, "Virtual Machine Allocation Policies against Co-resident Attacks in Cloud Computing," *Proc.IEEE International Conference on Communications (ICC 2014)*, pp. 786-792, 2014.

[25] Y. Han, J. Chan, T. Alpcan, and C. Leckie, "Using Virtual Machine Allocation Policies to Defend against Co-resident Attacks in Cloud Computing," *IEEE Transactions on Dependable and Secure Computing*,published on 4 May 2015, available from: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=7101258.

[26] Chaudhary, G., & Narahari, Y. (2013). A Stackelberg Game Approach for Secure and Efficient Surveillance. *Procedia Computer Science*, *24*, 205-216.

[27] "CloudSim," 2015; http://www.cloudbus.org/cloudsim/.

[28] 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.

[29] Khorshed, T., Shawkat, A.B.M., Wasimi, S. (2012) "Classifying different denial-of-service attacks in cloud computing using rule-based learning". Journal-Security and Communication Networks, volume 5, Issue 11 Pages 1235-1247.

[30] Bedi, H. S., & Shiva, S. (2012, August). Securing cloud infrastructure against co-resident DoS attacks using game theoretic defense mechanisms. In *Proceedings of the International Conference on Advances in Computing, Communications and Informatics* (pp. 463-469). ACM.

[31] Liu, H. (2010) "A new form of DoS attack in a cloud and its avoidance mechanism". In ACM Cloud Computing Security Workshop (CCSW), pages 65–76

[32] Han, Y., Alpcan, T., Chan, J., Leckie, C., & Rubinstein, B. I. (2016). A game theoretical approach to defend against co-resident attacks in cloud computing: Preventing co-residence using semi-supervised learning. *IEEE Transactions on information Forensics and Security*, *11*(3), 556-570.

[33] Qiu, Y., Shen, Q., Luo, Y., Li, C., & Wu, Z. (2017, August). A Secure Virtual Machine Deployment Strategy to Reduce Co-residency in Cloud. In *Trustcom/BigDataSE/ICESS, 2017 IEEE* (pp. 347-354). IEEE.

[34] Atya, A. O. F., Qian, Z., Krishnamurthy, S. V., La Porta, T., McDaniel, P., & Marvel, L. (2017, May). Malicious co-residency on the cloud: Attacks and defense. In *INFOCOM 2017-IEEE Conference on Computer Communications, IEEE*(pp. 1-9). IEEE.

[35] Yu, S., Gui, X., Lin, J., Tian, F., Zhao, J., & Dai, M. (2014). A security-awareness virtual machine management scheme based on Chinese wall policy in cloud computing. *The Scientific World Journal*, *2014*.

[36] Ruiz-Vanoye, J.A. and Díaz-Parra, O.: A Mosquito Swarms Algorithm for the Traveling Salesman Problem. Unpublished manuscript.

[37] Y. Azar, S. Kamara, I. Menache, M. Raykova, and B. Shepard, "Co-location-resistant clouds," in *Proc. 6th ACM Workshop Cloud Comput. Secur.*, 2014, pp. 9–20