J Inf Process Syst, Vol.13, No.3, pp.476~490, June 2017
https://doi.org/10.3745/JIPS.01.0015

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

NDynamic Framework for Secure VM Migration over Cloud Computing

Suresh B. Rathod* and V. Krishna Reddy*

Abstract

In the centralized cloud controlled environment, the decision-making and monitoring play crucial role where in the host controller (HC) manages the resources across hosts in data center (DC). HC does virtual machine (VM) and physical hosts management. The VM management includes VM creation, monitoring, and migration. If HC down, the services hosted by various hosts in DC can't be accessed outside the DC. Decentralized VM management avoids centralized failure by considering one of the hosts from DC as HC that helps in maintaining DC in running state. Each host in DC has many VM's with the threshold limit beyond which it can't provide service. To maintain threshold, the host's in DC does VM migration across various hosts. The data in migration is in the form of plaintext, the intruder can analyze packet movement and can control hosts traffic. The incorporation of security mechanism on hosts in DC helps protecting data in migration. This paper discusses an approach for dynamic HC selection, VM selection and secure VM migration over cloud environment.

Keywords

Cloud Computing, Datacenter (DC), Host Controller, Physical Host, Virtual Machine (VM)

1. Introduction

Virtualization is a default technology to address resources by using partitioning, isolation, and encapsulation. Virtualization helps cloud providers to deploy its resources on-demand. Virtual machine (VM) is a core element running on hypervisor consumes resources like CPU, memory, storage and bandwidth from physical hosts (PH) [1]. As per NIST (National Institute of Standards and Technology) [2], cloud enables access to a various shared pool of resources that includes networks, servers, storage as convenient to the user and on demand [3]. The minimum management efforts required to do deployment, provisioning and releasing resources [4]. As per NIST, cloud models can be of public, private, and hybrid.

Private cloud is solely created to provide organizational service requirement. Fig. 1 shows the overall architecture for virtualization. The resources in private cloud never get shared in between companies. The hardware resources are given to end user as the resource under control the firewall. Tools like OpenStack, openNebula used to create private.

Manuscript received August 18, 2016; first revision February 14, 2017; accepted April 11, 2017.
Corresponding Author: Suresh B. Rathod (sureshrathod1@gmail.com)
* Dept. of Computer Science and Engineering, KL University, Vaddeswaram, Guntur, India (sureshrahtod1@gmail.com, vkrishnareddy@kluniversity.in)

The resources sharing done to perform a specific task, security requirements, policy, and compliance considerations [4]. Here resource shared in a group of people, community people or organization [3].

Public cloud is a model that provides services to the end user, where cloud user can access services by signing an agreement with cloud provider [5]. Cloud provider has full control over DC and cloud user uses these services from the cloud provider. An example of these services includes Amazon, salesforce, etc. Service provider deploys his infrastructure across various countries to have the better accessibility of services. These services can be in the form of SaaS, PaaS, and IaaS. End user does access these services by signing service level agreement with the cloud provider [3].

Hybrid cloud is the cloud service model involves an association of public cloud, private cloud, and community cloud options. Here, the organization collaborates their services in public domain and maintaining accessibility to their infrastructure through the firewall or publically accessible via network [3].

The community cloud is a model where set of resources shared among several organizations [3] in support of specific people in a community [3]. These services shared for a specific task, security provisions, policy, and to fulfill agreement [3].

Public cloud is a model that provides services to the end user, where cloud user can access services by signing an agreement with cloud provider [5]. Cloud provider has full control over DC and cloud user uses these services from the cloud provider. An example of these services includes Amazon, salesforce, etc. Infrastructure of these service providers located all over the world. These services can be in the form of SaaS, PaaS, and IaaS. The services are offered to the cloud user in the form of SLA [3].

Hybrid cloud is the cloud service is an association of public cloud, private cloud, and community cloud options. Here, organization collaborates their services in public domain maintaining accessibility to various resources after installing firewall or publically accessible via network [3].

DC has thousands of hosts in a rack interconnected by gigabit networking components. Physical hosts in a DC connected with each other through high speed network. End user can be a single user to multiple users or it can be the companies. Cloud computing (CC) helps many organizations to reduce investment cost by adopting cloud and executing their jobs on VM instances [3]. Virtual Machine migration facilitates the end user to move VM instances across hosts in DC. DC has thousands of hosts in a rack interconnected by gigabit networking components. The Internet helps in connecting physical host in a DC to connect to the external world.

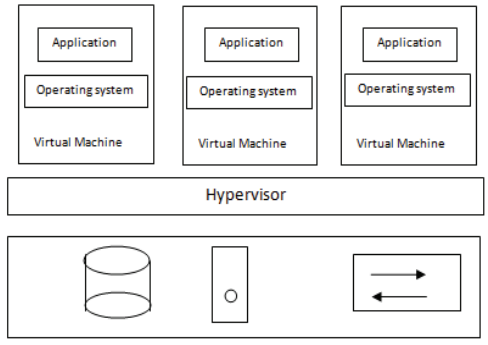

Fig. 1. Virtualization.

Each VM differs by resource it has and the type of job it is accomplishing. As a result, physical hosts in DC have many virtual machines running simultaneously and dissimilar task completion time. Each

VM has its own CPU design, OS, and various resources including disk, network, etc. Fig. 2 demonstrates an overall scenario to building data center.

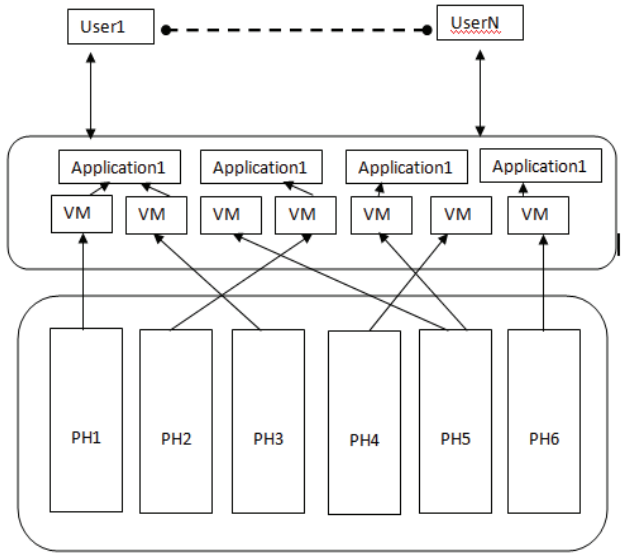

Fig. 2. Data center with physical host and virtual machines.

VM migration is an efficient technique involved in CC where resource provisioning involves selecting VM, migration, and VM placement on different PHs. VM placement uses approaches like linear programming, constraint programming, bin packing, ant colony algorithm and genetic algorithm [6]. VM placement consideration involves several aspects such as resource allocation, server consolidation, and energy consumption [7]. Traffic-aware VM placement algorithms proposed by [8] wherein they discussed how network scalability will be an improvement and what will be the network impact on architectures and traffic patterns for optimal VM placement. Several vulnerabilities still exist in Xen and KVM hypervisors for live VM migration implementation. The host provides VM kernel, application state and VM's sensitive data including users password, service accessing keys, etc., transmitted over the network in the form of clear text. Confidential data, untrusted platforms, VM provisioning and management, should considered. Hence, this research works.

2. Related Work

Cloud is gaining additional attention towards the infrastructure cost and DC efficiency since last decades to increase in popularity among users of CC. Frequent VM migration helps in increase availability of various services for a longer time. Live VM migration involves memory, state, and network migration. More work has been done in memory migration; where the state of VM is migrated along with memory. Memory migration for VM occurs as either per copy or post copy. The VM migration involves memory migration wherein VM memories along with its state from the source host to the destination host. More work has been done in memory migration; the migration involves transferring running VM state and VM's memory page migration.

The VM migration can be done in applying either in pre-copy or post-copy approach.

1) Post-copy migration: Here VM's memory contents are migrated towards destination hosts only when VM's processor state is migrated to the destination host [9].
2) Pre-copy migration: Here VM's state is migrated at end. The memory pages of the VM are transferred in rounds. In the last round VM state is transferred to the destination host [9].

Post-copy migration technique involves moving VM memory contents at first, processor state at the end [9]. Pre-copy migration technique, the processor state moved at last, the memory page associated with VM moved in rounds and in the last round VM state is transferred to destination host [9].

Both memory migration techniques have some pros and cons. The pre-copy approach does initial VM memory page migration and VM state last, this cause problem like, if a VM is write-intensive it creates dirty pages and migration time is equal to the sum of the time required for memory pages in each round of VM [10]. The time required might be uncountable if a VM is write-intensive else it is the sum of migration time for dirty pages in each round [11].

Thus, many researchers started to work on providing high-quality service along with providing security to cloud user data. In [12], the authors explained live and incremental migration techniques wherein they considered the migration with a Xen platform with their TPM three phase algorithm, termed as pre-copy, freeze and copy, and post copy. In [11], the authors have discussed how VM allocation policy with respect to physical resources such as CPU and Memory. The purpose was reducing the number of active hosts in DC.

In [13], author has discussed various solutions to provide secure VM migration towards end system.

In [14], the authors have discussed how to authorize VM towards end systems by adding hash code along with VM. Hypervisor uses own attribute parameter to check whether a VM is malfunctioning.

If the hypervisor compromised, all resources including VM are accessible to the attacker. An attacker can launch by gaining access to either VM or PH. The VM and PH can act as a launch pad for attacker where an attacker can use a compromised VM to launch an attack on remaining hosts in a cloud [15]. In [16], author has explained a mechanism to provide monitoring and reliability check model. Here author discussed HSEM component for monitoring each VM behavior whereas HAREM checks reliability for a host. In [17], the authors have discussed hypervisor-based security wherein they explored the mechanism for secure booting with several approaches for secure I/O calls occurred across the hypervisor and guest OS.

In [18], the authors had presented a decentralized solution termed as DAM, where they discussed how PH can reorganize itself as per the mentioned policy towards infrastructure layer and software layer. Several cloud providers like Google, Amazon, HP, and IBM have central or decentralize architecture incorporated to provide uninterrupted service to cloud user. Cloud providers like Amazon they have incorporated central architecture to provide AWS services without fail. Amazon's elastic watch is a functional utility that does record activities for EC2 CPU, disk, and network and raises the alarm in failure [19]. One of current instance occurred on September 22, 2015 where AWS services stopped working because of failure occurred on S3 instance [4]. AWS architecture currently supports central architecture [4]. Consistent failures in services lead to increase in downtime. The downtime avoidance needs new architectural modification.

The mechanism that controls and monitors services for hosts in DC need decentralized architecture.

The decentralized architecture proposed by the authors in [20] of their research paper where they proposed how the peer to peer distributed network is useful to do decentralized VM migration wherein they explained how nodes in DC forwards its own CPU utilization to itself and all other nodes in DC [10].

The mechanism proposed in [21] deals with threshold based VM selection policy based on upper and lower threshold value. If a VM migration to another host crosses an upper limit to the destination host, a VM that causes an increase in CPU utilization towards destination host needs to be re-migrated again. The best mechanism is to identify host with a minimum threshold value and worst case is in an infinite loop if unable to find suitable hosts, leading a DC to an inconsistent state.

3. Proposed Method

This section deals with proposed framework having HC selection, VM selection, and secure VM migration. Framework formation involves establishing a network connection in physical hosts. Each host needs configuration with the necessary tools to create a cloud platform in DC. Host configuration done with Xen, KVM, or VMWARE hypervisor. Each host in DC configured such that it has its own table where it maintains hosts details information including its address, the number of VM hosted on it, what is CPU utilization and the destination host address (HC) to whom it forwards its detail and a flag specifying whether it is acting as HC or CH. The message structure associated with each host (CH) to make communication with HC has the following format.

Host address	No of VM	CPU utilization	Flag	Destination host address

3.1 Host Controller Selection

Every host has the same configuration as the controlling host. The proposed mechanism consists of selecting the hosts from several hosts in DC that do VM provisioning and management. Initially, a random host selected as a HC that continuously monitors hosts in DC. Host monitoring done by traversing the set of information received from each host in DC. Decisions for VM migration initiation by HC done by maintaining a special table wherein it keeps all hosts received information along with own information. Host in DC starts sending its own information in an above-mentioned message format to the current HC at after specified interval. Each host configured with a daemon thread such that it comes in running whenever a controller host will act as a HC. This daemon thread initiates the procedure to locate the host with minimum CPU utilization and maximum CPU utilization by traversing each row record from a table associated with HC. HC starts comparing its own detail with all hosts detail and marks a host as a CH as new wherein CH has minimum CPU utilization. Marking a host (CH) as new HC, the old HC does send a ping message to new HC to check whether the new HC is alive. HC starts comparing its own detail with all hosts detail by looking up the table wherein all hosts' information stored and marks the corresponding CH as new HC. When CH identified the old host sends ping message to check the marked host status whether it is alive. Finding new HC splits into two cases:

1) If old HC doesn't receives reply within specified time it initiates new HC selection procedure .

2) If old HC receives reply within specified time from new HC, it broadcasts next HC address to all host entries present in old HC table.

The message structure associated with each HC to do communication with CH has the following format.

Src address	New HC address	Old HC address

On receiving new HC address each host updates its own table information and starts transmitting their detail to new HC. New HC initiates the new HC selection after a fixed time interval. This process continues until hosts in DC are active. The random host CH-2 selected and marks as HC. After the certain interval, it starts receiving messages from all remaining hosts, including CH-2, CH-1, CH-3, CH-4, CH-5, CH-6, and CH-7. CH-2 stores all host records. Host CH-2 starts traversing host's details to find a host with minimum CPU utilization and minimum VM's running. As in Table 1, the CH-2 will find a CH-4 as new HC as it has a minimum CPU utilization as compared with all remaining hosts in DC and CH-3 as maximum CPU utilization.

The CH-2 will mark CH-4 as new HC and does check whether it is active. If it is active, CH-2 will update destination address in its own table and broadcast CH-4 host's address to all CH-1, CH-2, CH-3, CH-4, CH-5, CH-6, and CH-7. The host detail for new HC will be as in Table 2.

Table 1. CPU utilization statistics for host in DC

Host name	CPU utilization	Current (HC)	Active VM's
CH-1	0.81	0	3
CH-2	0.62	1	2
CH-3	0.96	0	3
CH-4	0.51	0	2
CH-5	0.83	0	3
CH-6	0.8	0	3
CH-7	0.8	0	3

Table 2. CPU utilization statistics for host in DC (host details for new HC)

Host name	CPU utilization	Current (HC)	Active VM's
CH-1	0.81	0	3
CH-2	0.62	0	2
CH-3	0.96	0	3
CH-4	0.51	1	2
CH-5	0.83	0	3
CH-6	0.8	0	3
CH-7	0.8	0	3

HC calls an HC selection algorithm and marks one of the PH as new HC by applying minimum CPU utilization and running VMs. Every host revises HC address and starts forwarding updated data to new HC. The below algorithm discusses HC selection.

Algorithm_HCSelection ()

```
1)    If flag==false
2)    {       DC_configure=true
3)            flag=true;
4)    else
5)            loc=random();
6)    for  i  =  1 to n
7)            Host[i].configuration = true
8)    if (i==loc)
9)            HOSTS[loc] = HC;
10)           break;
11)   for  i =  1 to n
12)           HOSTS [i]. address = HOSTS [loc].address;
13)   for i=0 to n
14)           if (loc == i)
15)           {
16)                   for  j = 0 to n
17)                   HOSTS[i].cpu[j]=HOSTS[i].cpu;
18)                   HOSTS[i].nVM[j]=HOSTS[j].VM;
19)           }
20)   for i =0 to n
21)   {
22)           If (HOSTS[i].util < curr_limit && HOSTS[i].cpu <curr_limit )
23)            HOSTS[i].n_HC =true;
24)             Break;
25)   }
```

3.2 Dynamic VM Selection

Each host has many virtual machines each with the different application deployed by user workload with varying CPU utilization. CPU utilization of VM changes as per change in workload. Physical host has limited resources including CPU, RAM, disk and network bandwidth, network, etc. Some of running VM instances consumes additional CPU power causing a host to down lead to all services hosted on such host down. Migration initiated for VM consuming additional CPU power. Migrating such VM from one of the hosts, a VM selection procedure needs to consider, that will find a VM consuming huge CPU power towards underlying host. The VM selection procedure involves receiving host detail from all hosts in DC to current controller host (HC). HC finds a host with minimum resource utilization and host with maximum resource utilization. When the procedure for identification of source hosts and destination host completed, HC initiates the trigger for VM migration on source host specifying destination with minimum CPU utilization. Fig. 3 demonstrates an overall architecture for VM selection and migration procedure.

The architecture has several physical hosts like CH-1, CH-2, CH-3, CH-4, CH-5, CH-6, and CH-7. Each host interconnected by networking components like gigabit switch and gigabit Ethernet cable.

On booting DC one of the hosts (controller) selected and marked as an HC. Here host with id 2 selected as HC. After receiving workloads from each host, the HC, here CH-2 checks its own CPU utilization and remaining host's CPU utilization and initiates trigger for VM selection. Here, CH-2 initiates trigger for VM selection and it finds the host with maximum CPU utilization and number of active VM instances on every host.

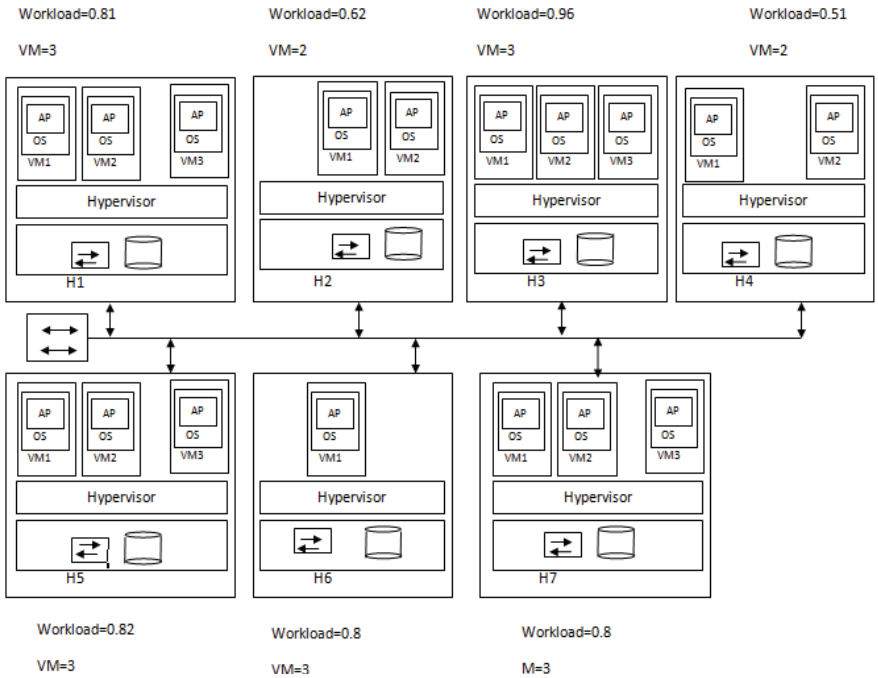

Fig. 3. Dynamic VM selection.

The algorithm for VM selection is as below.

Algorithm_VMselction()

1) Calculate workload ()
2) for (each Hosts in HostList)
3) {
4) if((HostList[Host].workload<HostList[Host+1].workload)&&(HostList[Host].vms<hostList [Host+1].vms))
5) { HostList[Host].dst=true
6) return 0;
7) }
8) else((HostList[Host].workload>HostList[Host+1].workload)&&(HostList[Host].vms>HostList [Host+1].vms))
9) HostList.[Host].src=true;
10) Return 0;
11) }

3.3 Secure VM migration

Many hypervisors like Xen, VMware, KVM, and Hyper-V supports VM migration. Migration can be within DC or in different DC. Hosts in same DC interconnected with each other through switches. Normal VM migration hosts share common storage accessed by NAS [6,22]. VM migration considers VM's RAM and data stream associated with tasks running or resource consumption by running process in VM. A stream of data contains sensitive and confidential user data. Host machines traffic and VM's traffics eparated using tags in packets moving from or to VM [23]. An intruder can identify traffic associated with VM by observing tag fields and might misuse VM. Attacker might do searchingwhether any host in DC is having misconfigured VLAN [24]. If an attacker finds such host, he can launch VLAN hopping attack [20] and can launch Man in middle attack by either using ARP spoofing or by using DNS poisoning [25].

The data in migration achieved direct live migration moves in the form of plaintext. Plaintext data encrypted with suitable ciphers likeAES, DES, 3DES, or MD5. The ssh with support of cipher enabled by uncommenting ciphers line from sshd_config file. The initial authentication for hosts achieved using RSA or Diffie-Hellman key exchange. Initial authentication done with RSA, where public and private keyshared with hosts in communication. The tunnel in Libvirt created by using qemu+ssh://url/system.

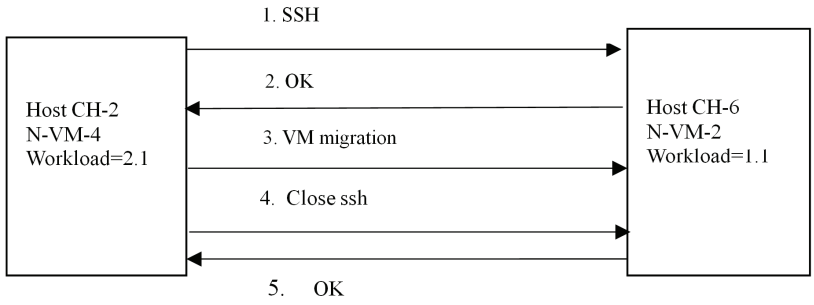

Fig. 4. Secure VM migration.

The VM migration with ssh protocol described in Fig. 4 where host CH-2 initiates establishing a secure connection by using ssh with host CH-6. Host CH-6, if it is active gives ok reply and opens a port for CH-2. If host CH-2 receives ok, it starts pushing VM's data and memory on a specified port to host CH-6. When VM migrated to host CH-6, host CH-2 does a request to close the connection with host CH-6. After receiving close request it closes port opened with host CH-6. Here the migration done by using RSA key exchange algorithm and AES cipher. The algorithm for VM migration is as below.

Algorithm Migration ()

1) HCSelection ()
2) if flag==false
3) No VM to migrate
4) else

```
5)  for i =0 to n
6)      if (HOSTS[i].util > up_threshold)
7)          src_victim = HOSTS[i].address
8)      if (HOSTS[i].util < low_threshold)
9)          dest_victim = HOSTS[i].address
10) Connect src_vivitm to dest_victim
11) Establish secure channel
12) Migrate (HOSTS[i].VM, src_victim, dest_victim)
13) End
```

4. Performance Evaluation

In this section, we will discuss the implementation of proposed system and comparison with previous work.

4.1. Implementation of Proposed System

The implementation of proposed system done by considering 7 HP Intel core i5-SS CPU 3.00 GHz with 500 GB HDD and 4 GB RAM. The RAM allocated to VM is 1024, 1 GB disk size, and 1vCPU. A proposed system configured with KVM/QEMU hypervisor. JDK 1.6.0 considered as software platform for implementation. All hosts in consideration connected by forming peer to peer network topology. Each host in DC configured such that they can act as both HC and CH. Initially, random host here CH-2 considered as HC. Table 2 illustrates the initial configuration associated when all host connected to the HC. Whenever user wants to see the details of controller host he need to provide his credentials to the server acting as controller host. Fig. 5 shows the window to connect controller host.

The host details displayed upon validating, credentials provided by the user. Fig. 6 illustrates displaying the current host and remote hosts. Fig. 6 has three blocks, first block shows NIC connection to the host (No), the number of virtual bridges available with the host. The second block shows underlying hypervisor details on physical hosts. Physical host has the following details, what is local host name, what hypervisor version installed on the host, maximum vCPU on underlying host and the URI to get access to the host. Third block illustrates VM instances on the host, the name of selected VM and id assigned by the underlying host and the state of the VM.

Live VM migration done by either shared or direct approach. Direct migration uses copying VM's disk image from one hosts storage to another host, and takes larger time as of shared migration. Direct migration is useful for disk of small or medium size. For larger disk direct disk storage can't considered. Zero down time for virtual machine obtained by considering shared migration. The shared disk implementation is either with SAN or NFS. In this we are considering NFS as shared approach, wherein the disk image of virtual machines available on /var/lib/Libvirt/images. This shared resource shared among CH-1, CH-2, CH-3, CH-4, CH-5, CH-6, CH-7. When HC identifies CH with minimum CPU utilization and maximum CPU utilization, this initiates migration, here as in Table 2 CH-4 finds CH-3 as host with maximum CPU utilization it selects one VM and migrates that to the CH-3.

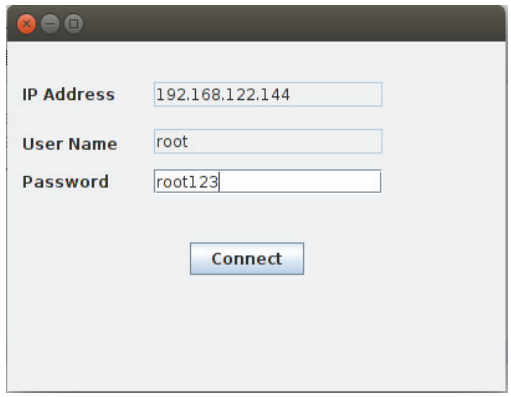

Fig. 5. Login to remote host.

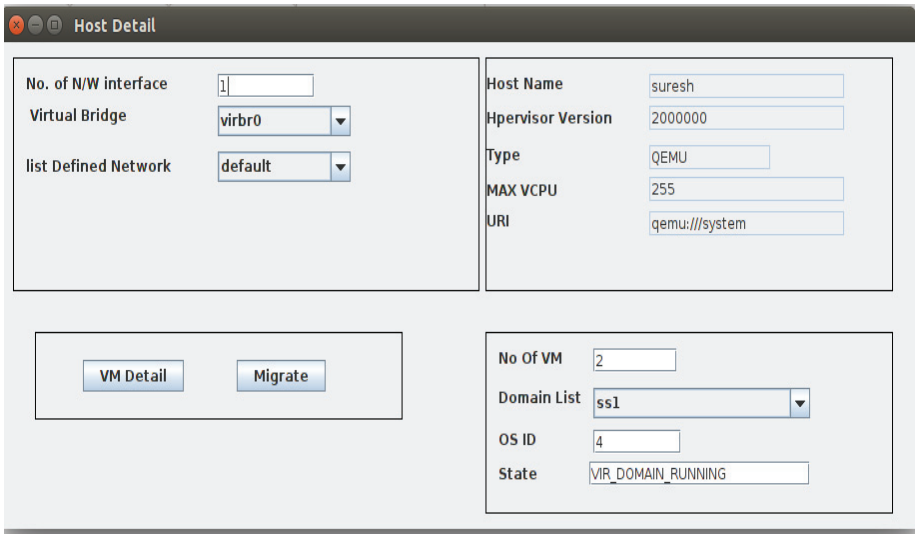

Fig. 6. Displaying remote and local host details.

4.2 Comparison with Existing Work

Table 3 illustrates the different framework formed for VM migration in CC environment. The different authors have considered VM migration either based on resource vector, CPU utilization, or SLA as base criteria but they have not discussed the security concern in VM migration that needs to be considered while migrating a VM from hosts in DC.

In summary, the proposed architecture succeeds in avoiding central failure and provides secure VM migration. As the proposed architectures evaluation is done by using NFS across hosts in DC helps to provide zero downtime in migration between hosts. The proposed architecture also helps in balancing the load across hosts in hosts in DC. The migration of VM is done based on the number of VM running instances and CPU utilization. It also helps in green computing by migrating VM instance from maximum CPU consumption host.

Table 3. Centralized and decentralized secure architectures

Sr. no.	Ref.	Approach utilized	Based on		Architecture		Host location	VM security	Access control
			S/W	H/W	Central	Distributed			
1	[26]	Intel vPro technology to provide trust service to software program		Yes	Yes				
2	[27]	Framework for secure VM migration role based access control policies to protect against unauthorized usage of migration privileges	Yes		Yes			Yes	Yes
3	[28]	Threat based security enforcement model using cryptography	Yes		Yes			Yes	
4	[29]	Considered hypercube based VM placement, migration.	Yes			Yes			
5	[30]	Considered p2p VM migration	Yes			Yes			
6	[31]	The approach is based on secure migration by using host based firewall as well as network firewall rules	Yes		Yes			Yes	Yes
7	[29]	Proposed an improved secure vTPM migration protocol	Yes		Yes			Yes	
8	[32]	Utilized three modules: data protector for data encryption and decryption; metadata manager for marshaling; and security guard live migration	Yes					Yes	
9	[33]	Uses firewall rule for source host and destination host authentication	Yes		Yes			Yes	

4. Conclusions

In this paper, we have proposed a decentralized secure virtual machine migration framework where the host categorized as controller host or controller host depending on workload on each host. The HC does decision for VM provisioning and VM management. The controller host sends host details to HC. This categorization of hosts helps in avoiding single point of failure and at the same time using the tunnel in migration avoids tampering of packets in VM in migration.

Acknowledgement

We would like to convey our thanks and gratitude towards the Head of Department Computer Science and Engineering, all the staff members of KL University who have been a source of inspiration in doing this research work.

References

[1] J. S. Reuben, "A survey on virtual machine security," in *Proceedings of TKK T-110.5290 Seminar on Network Security*, 2007.

[2] H. Jin, W. Gao, S. Wu, X. Shi, X. Wu, and F. Zhou, "Optimizing the live migration of virtual machine by CPU scheduling," *Journal of Network and Computer Applications*, vol. 34, no. 4, pp. 1088-1096, 2011.

[3] D. S. Dias and L. H. M. Costa, "Online traffic-aware virtual machine placement in data center networks," in *Proceedings of Global Information Infrastructure and Networking Symposium (GIIS),* Choroni, Venezuela, 2012, pp. 1-8.

[4] V. Mann, A. Kumar, P. Dutta, and S. Kalyanaraman, "VMFlow: leveraging VM mobility to reduce network power costs in data centers," in *Proceedings of the 10th International IFIP TC6 Conference on Networking*, Valencia, Spain, 2011, pp. 198–211.

[5] H. Jin, S. Ibrahim, T. Bell, W. Gao, D. Huang, and S. Wu, "Cloud types and services," in *Handbook of Cloud Computing*. New York, NY: Springer, 2010, pp. 335-355.

[6] U. Deshpande and K. Keahey, "Traffic-sensitive live migration of virtual machines," in *Proceedings of 2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, Shenzhen, China, 2015, pp. 51-60.

[7] R. Narayani and W. A. Banu, "Framework for provenance based virtual machine placement in cloud," *International Journal of Education and Management Engineering*, vol. 5, no. 1, pp. 19-26, 2015.

[8] X. Meng, V. Pappas, and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," in *Proceedings of INFOCOM*, San Diego, CA, 2010, pp. 1–9.

[9] H. Mi, H. Wang, G. Yin, Y. Zhou, D. Shi, and L. Yuan, "Online self-reconfiguration with performance guarantee for energy-efficient large-scale cloud computing data centers," in *Proceedings of 2010 IEEE International Conference on Services Computing (SCC),* Miami, FL, 2010, pp. 514-521.

[10] National Institute of Standards and Technology (NIST) cloud computing program [Online]. Available: http://www.nist.gov/.

[11] P. Xiao, Z. Hu, D. Liu, G. Yan, and X. Qu, "Virtual machine power measuring technique with bounded error in cloud environments," *Journal of Network and Computer Applications*, vol. 36, no. 2, pp. 818-828, 2013.

[12] Y. Luo, B. Zhang, X. Wang, Z. Wang, Y. Sun, and H. Chen, "Live and incremental whole-system migration of virtual machines using block-bitmap," in *Proceedings of 2008 IEEE International Conference on Cluster Computing*, Tsukuba, Japan, 2008, pp. 99-106.

[13] D. Perez-Botero, "A brief tutorial on live virtual machine migration from a security perspective," University of Princeton, Princeton, NJ, 2011.

[14] C. Li, A. Raghunathan and N. K. Jha, "A trusted virtual machine in an untrusted management environment," *IEEE Transactions on Services Computing*, vol. 5, no. 4, pp. 472-483, 2012.

[15] J. Dong, X. Jin, H. Wang, Y. Li, P. Zhang, and S. Cheng, "Energy-saving virtual machine placement in cloud data centers," in *Proceedings of 2013 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid),* Delft, Netherlands, 2013, pp. 618-624.

[16] F. Sabahi, "Secure virtualization for cloud environment using hypervisor-based technology," *International Journal of Machine Learning and Computing*, vol. 2, no. 1, pp. 39-45, 2012.

[17] Y. Cheng and X. Ding, "Guardian: hypervisor as security foothold for personal computers," in *Trust and Trustworthy Computing*. Heidelberg: Springer, 2013, pp. 19–36.

[18] X. Chen, X. Gao, H. Wan, S. Wang, and X. Long, "Application-transparent live migration for a virtual machine on network security enhanced hypervisor," *China Communications*, vol. 8, no. 3, pp. 32-42, 2011.

[19] E. Feller, C. Morin, and A. Esnault, "A case for fully decentralized dynamic VM consolidation in clouds," in *Proceedings of 2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom),* Taipei, Taiwan, 2012, pp. 26-33.

[20] T. Hirofuchi, H. Nakada, S. Itoh, and S. Sekiguchi, "Reactive consolidation of virtual machines enabled by postcopy live migration," in *Proceedings of the 5th International Workshop on Virtualization Technologies in Distributed Computing*, San Jose, CA, 2011, pp. 11-18.

[21] X. Wang, X. Liu, L. Fan, and X. Jia, "A decentralized virtual machine migration approach of data centers for cloud computing," *Mathematical Problems in Engineering*, vol. 2013, article no. 878542, 2013.

[22] W. F. Hsu, G. H. Luo, S. M. Yuan, and C. T. Tsai, "Constructing private cloud storage using network attached storage," in *Proceedings of 2012 9th International Conference on Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing (UIC/ATC)*, Fukuoka, Japan, 2012, pp. 713-718.

[23] R. Delgado, "The need for decentralized cloud computing," 2015 [Online]. Available: https://www.socpub.com/articles/the-need-for-decentralized-cloud-computing-14741.

[24] Amazon Web Services, "AWS Well-Architected Framework," October 2015 [Online]. Available: https://aws.amazon.com/ko/blogs/aws/are-you-well-architected/.

[25] D. Diaconescu, F. Pop, and V. Cristea, "Energy-aware placement of VMs in a datacenter," in *Proceedings of 2013 IEEE International Conference on Intelligent Computer Communication and Processing (ICCP),* Cluj-Napoca, Romania, 2013, pp. 313-318.

[26] M. Mukhtarov, N. Miloslavskaya, and A. Tolstoy, "Network security threats and cloud infrastructure services monitoring," in *Proceedings of 7th International Conference on Networking and Services*, Venice/Mestre, Italy, 2011, pp. 141-145.

[27] A. Shribman and B. Hudzia, "Pre-copy and post-copy VM live migration for memory intensive applications," in *European Conference on Parallel Processing*. Heidelberg: Springer, 2012, pp. 539-547.

[28] M. Nanavati, P. Colp, B. Aiello, and A. Warfield, "Cloud security: a gathering storm," *Communications of the ACM*, vol. 57, no. 5, pp. 70-79, 2014.

[29] M. Pantazoglou, G. Tzortzakis, and A. Delis, "Decentralized and energy-efficient workload management in enterprise clouds," *IEEE Transactions on Cloud Computing*, vol. 4, no. 2, pp. 196-209, 2016.

[30] D. Loreti and A. Ciampolini, "A decentralized approach for virtual infrastructure management in cloud datacenters," *International Journal on Advances in Intelligent Systems*, vol. 7, no. 3/4, pp. 507-518, 2014.

[31] N. Ahmad, A. Kanwal, and M. A. Shibli, "Survey on secure live virtual machine (VM) migration in Cloud," in *Proceedings of 2013 2nd National Conference on Information Assurance (NCIA),* Rawalpindi, Pakistan, 2013, pp. 101-106.

[32] G. Booth, A. Soknacki, and A. Somayaji, "Cloud security: attacks and current defenses," in *Proceedings of 8th Annual Symposium on Information Assurance (ASIA'13)*, Albany, NY, 2013, pp. 4-5.

[33] S. Berger, R. Caceres, K. A. Goldman, R. Perez, R. Sailer, and L. van Doorn, "vTPM: virtualizing the trusted platform module," in *Proceedings of 15th Conference on USENIX Security Symposium*, Vancouver, Canada, 2006, pp. 305-320.

Suresh B. Rathod http://orcid.org/0000-0002-6772-1529

He has completed M.E. from the University of Pune, Maharashtra. He has completed his B.E. from Dr. BAMU Aurangabad, Maharashtra. His research area is cloud computing. Since January 2014, he is with KL University of Computer Science and Engineering from Vijayawada, Andhra Pradesh, India as a PhD candidate.

V. Krishna Reddy

He is presently Professor in the Department of Computer Science & Engineering, KL University-Vijayawada, Andhra Pradesh, India. He received a Ph.D. degree from Acharya Nagarjuna University, Guntur, Andhra Pradesh. His research interests include cloud computing, network security, and data mining. He published more than 34 papers in refereed international journals and 15 papers in conferences. He is an active member of ACM, ISTE and Computer Society India (CSI).