

Thesis Report
on

A Framework for Secure Container Migration with Blockchains in Virtualized Cloud Infrastructure

Dual Degree Engineering
(Computer Science and Information Technology)

Submitted by
Sourav Kumar Panda
Regd No.1391078010
Branch. DDE(CSIT)

Under the guidance of
Mr. Jyoti Prakash Sahoo
Assistant Professor
Dept. of CS&IT



Department of Computer Science and Information
Technology
Institute of Technical Education and Research, BBSR
Siksha 'O' Anusandhan University

2018



INSTITUTE OF TECHNICAL EDUCATION & RESEARCH,
BHUBANESWAR

CERTIFICATE

This is to certify that the Thesis Report entitled “**A Framework for Secure Container Migration with Blockchains in Virtualized Cloud Infrastructure**” is a bonafide work done by **Sourav Kumar Panda** bearing Regd. No. **1391078010** of 10th semester, **Department of Computer Science & Information Technology**, towards the completion and award of **Dual Degree Engineering** from **ITER**, under **Siksha 'O' Anusandhan University Odisha**.

HOD :

Dr. Ajit Kumar Nayak
Professor & HOD
Dept. of CS&IT

Guide :

Mr. Jyoti Prakash Sahoo
Assistant Professor
Dept. of CS&IT



DECLARATION

I declare that every parts of this Thesis Report is genuinely my work and has not been submitted for any other degree. I acknowledge that if any sort of malpractice is detected in relation to this report I shall be held liable for it.

Submitted by

Sourav Kumar Panda

Regd No. 1391078010

DDE(CSIT)

Dept. of CS&IT



ACKNOWLEDGEMENT

I avail this opportunity to express my profound sense of gratitude to Mr. Jyoti Prakash Sahoo, Assistant Professor, Dept. of CS&IT, Dr. Kaberi Das, Associate Professor, Dept. of CSE and Dr. Ajit Kumar Nayak, Professor & Hod, Dept. of CS&IT for their constant supervision, guidance and encouragement right from the beginning till the completion of the thesis.

I wish to thank all the members of faculty of CSIT department for their valuable advice and encouragement. Last but not the least a special thanks to my parents and all my colleagues for their support and compassion.

Sourav Kumar Panda
Regd No. 1391078010
DDE(CSIT)

Abstract

Cloud Computing is one of the emerging technology which is broadly used throughout the world is having high impact on current IT scenario. Cloud Computing is supported with large-scale Virtualized Data centers behind the scene. The Virtualization techniques allows the users to interact with the services irrespective of the underlying Infrastructure of Cloud Data centers. Virtual Machines and Containers are used in Cloud Infrastructure to efficiently provide the services to the end users. The Cloud Data centers adopts Consolidation strategies like Application Consolidation, I/O Consolidation and Server Consolidation in order to ensure efficient use of computer resources. Server Consolidation allows more than one servers to reside on Physical Server in order to utilize all of a server's available resources and prevents the Server as well as Storage for getting underutilized. Server Consolidation is done over the Data centers to achieve better performance and enhances the service quality where Migration plays an key role. Meanwhile, it is observed that there may be a threat of malicious attack of intruders during Migration process. Blockchain, as one of the prominent technology is overcoming the existing technologies in security aspect. Blockchain technology is used to achieve robust and secure migration process in Data Center. To gain a deeper understanding of Blockchain based Migration process, the practical part of the thesis demonstrates the secured working model.

Keywords:- Cloud computing; Virtualization; Virtual Machines; Containers; Dynamic Consolidation; Migration; Blockchain, Blockchain based Container Migration.

Contents

1	Introduction	1
2	Background	2
2.1	Cloud Computing	2
2.2	Virtualization	5
2.2.1	Virtualization Techniques	6
2.2.2	Hypervisors	7
2.3	Virtualization with Containers	8
2.4	Containerization vs Virtualization	8
3	Virtual Machines Consolidation	9
3.1	Static VM consolidation	10
3.2	Dynamic VM consolidation	10
4	Virtual Machines Placement	12
4.1	VM Selection Policy	12
4.2	VM Placement Policy	14
5	Containers	19
5.1	Container Implementations	20
5.2	Containers in action	25
5.3	Container PLacement	27
6	Blockchain based Container Migration	32
6.1	Blockchain	32

6.2	Blockchain Working Model	33
6.3	Literature survey for Blockchain based migration . .	34
6.4	Proposed Model	34
6.5	Implementation	35
7	Conclusion	42

List of Figures

2.1	NIST definition of Cloud Computing	3
2.2	Full Virtualization vs Para Virtualization	6
2.3	Bare Metal Hypervisors vs Hosted Hypervisors	7
6.1	Proposed model	35
6.2	Blockchain with linked Blocks	38
6.3	Valid Blockchain	39
6.4	Invalid Blockchain	41

List of Tables

5.1	Comparision of Containers	24
5.2	Comparisions of Algorithm and Techniques in Virtualization	30
5.3	Comparisions of Algorithm and Techniques in Containerization	31
6.1	Blockchain importance on Container migration . . .	33

Chapter 1

Introduction

The first chapter starts with description of the thesis with its outline. The second chapter animates the background details of Cloud Computing, along with the essential characteristics, services and deployment models. Virtualization, the key feature of Cloud Computing is introduced with its various implementation techniques and different methods to achieve it. Container-based Virtualization which is widely known as Containerization is introduced. Lastly, Why Container is preferred over Virtual machines is discussed.

Next, the focus shifts to the Virtual Machine Consolidation, presenting Static and Dynamic VM Consolidation. Meanwhile, the problem of Dynamic Consolidation is described. The VM Placement technique including VM Selection Policy and a literature survey on VM Placement Policy is projected. The fifth chapter introduces Containers and its key terms. Besides, it broadcasts where the implementation of container is done. Moreover, an inquisitive study of containers trending action is delineated followed by a survey on Container Placement.

Finally, in Blockchain based Container Migration, Blockchain with its working model is discussed. Additionally, a survey based on container migration and Blockchain based migration was done and built up a Proposed model to secure the container migration with Blockchain technology. Lastly, implemented the proposed model and extended a new secured methodology for container migration with Blockchain.

Chapter 2

Background

2.1 Cloud Computing

Cloud Computing is a tensile,worthwhile, and act as a platform which provide consumer IT services throughout the Internet[22].It is found to be one of the prominent techniques which is adopted all around the world.Cloud computing focus on providing information as a service by centralizing the computing resources that are connected in the network.Cloud,is termed the the network which shares the resources.

It is also known as flexible computing.Here,the Cloud Service Providers assures the Cloud Service users to allow accessing the resources up to their need,by expanding and shrinking the resources at instances,and pay on the basis of resources usage under Service Level Agreement(SLA)[25].

According to The National Institute of Standards and Technology,Cloud Computing is defined as a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources(e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction[28].

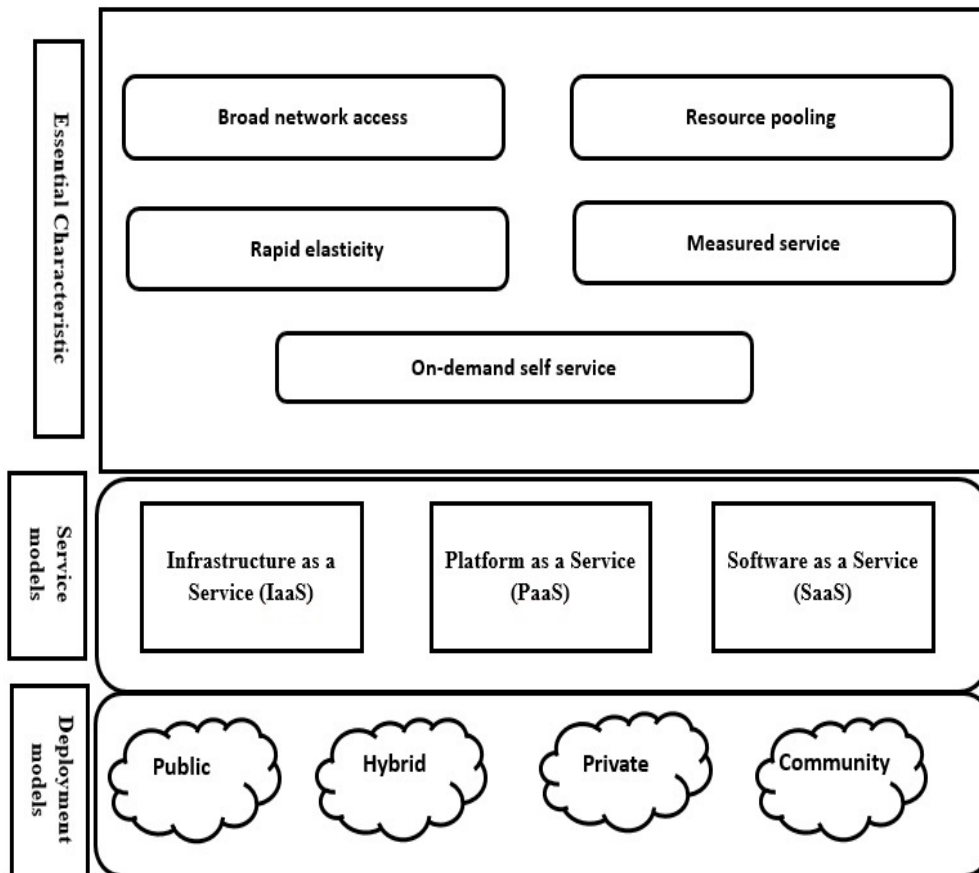


Figure 2.1: NIST definition of Cloud Computing

Cloud Computing provides five essential characteristics. The are as follows:-

- On-demand self-service:- Without any Human interaction with the service provider the consumer can automatically provisioned the computing capabilities.
- Broad network access:- All the computing capabilities that are available throughout the network can be accessed through standard mechanism by the customers.
- Resource pooling: All the computing resources are pooled together which are assigned to the customer depending upon their

demands.

- Rapid elasticity: Depending upon the consumer demand resources are assigned and are released after the work get completed.
- Measured service: Total usage of resources is monitored and recorded for the customer and the provider for the utilized service.

The service provided by the cloud is categorized into software, platform and infrastructure which are delivered to the client devices over the network. They are as follows:-

- Software as a Service:- The consumer can use the application of the provider running in the cloud infrastructure. The customer does not handle the underlying cloud infrastructure like storage, network, server etc.
- Platform as a Service:- The consumer is provided with a platform which can be run in any system and environment where the application can be developed, can be tested and deployed in the cloud.
- Infrastructure as a Service:- The consumer can use the resources to deploy and run the software including the OS and the application.

Cloud Deployment models are as follows:-

- **Public Cloud:-** The cloud environment where a third-party provides the resources to the clients through the network.
- **Private Cloud:-** A single organization exclusively uses all the cloud infrastructure.
- **Hybrid Cloud:-** It is the composition of two or more different cloud infrastructure i.e (public,private,community).
- **Community Cloud:-** The community of the consumers of an organization with same concern exclusively use the cloud infrastructure.

2.2 Virtualization

Virtualization is a process of abstracting physical resources such as storage,network and compute from the physical machine and creates virtual resources(eg. Virtual Storage,Virtual Network and Virtual Compute).Virtualization enables different Virtual Machines to run on a Physical compute system and isolate each Virtual machine from others and shares the host operating System.

Using Virtualization,developing software became easy as it allows hardware abstraction and server consolidation which turn a physical system into numbers of virtualized system which effectively reduce the cost as well as the energy. Various Virtual machine technologies are VirtualBox,Parallels Desktops,QEMU and Bootcamp.

Virtualization is achieved by Hypervisors.Hypervisors are categorized into two types:bare-metal and hosted.Likely OS,a bare-metal hypervisors is directly installed in the physical compute hardware and the Hosted hypervisors is installed as an application on the Operating System,lagging direct access to the physical compute resources.

2.2.1 Virtualization Techniques

The Hypervisors can be implemented using different virtualization techniques. They are as follows:-

- Full virtualization:- The Hypervisors produce an illusion to the virtual machines that the hardware is dedicatedly used by the virtual machine, which is done by trapping the instructions between the virtual machines and the host. So, the instruction of the virtual machine access the host hardware like its own resources and which make the Vm to think that no other guest virtual machine is running in the system.
- Para virtualization:- Here the Virtual machine is aware about others guest virtual machine running in the virtualized environment.
- Nested Virtualization:- It is a process where the hypervisors allow another hypervisors and its virtual machines to run inside it.s

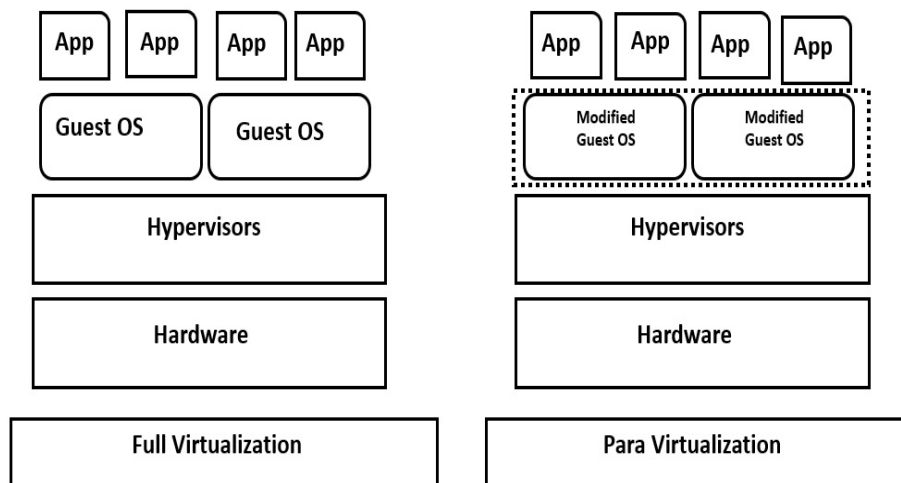


Figure 2.2: Full Virtualization vs Para Virtualization

2.2.2 Hypervisors

Virtualization allows different instances of Virtual Operating System to run on a Single host Operating System. The Virtualization is achieved with the help of Hypervisors or Virtual Machine Manager. It is classified into :-

- **Bare Metal Hypervisors:-** These Hypervisors runs directly on top of the System hardware so it directly access the hardware resources and is responsible to allocate the resources to the guest Virtual Machines running in the system. Xen, Hyper-V and VMware ESX are some of the bare metal hypervisors.
- **Hosted Hypervisors:-** These hypervisors run as an application in the host Operating System. As compared to the bare metal it has less hardware issues as the Operating system directly interact with the host hardware. VirtualBox and VMware Workstation are some of the Hosted Hypervisors.

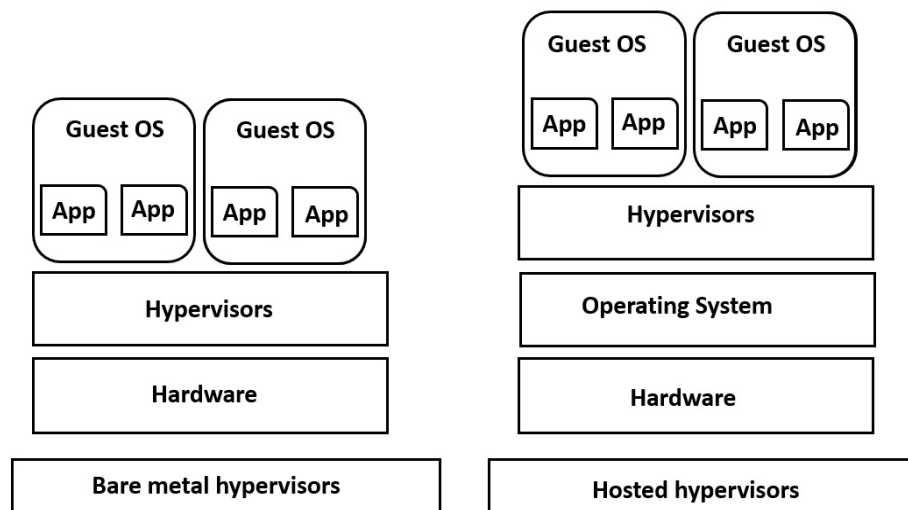


Figure 2.3: Bare Metal Hypervisors vs Hosted Hypervisors

2.3 Virtualization with Containers

Containerization is also known as container-based virtualization which provide light weighted compact runtime, is capable to check, develop and deploy applications into several servers and capable of connecting the containers[31]. Containers are light weight operating system that runs on the host system. It allows instructions native to the core CPU to execute and eliminates the requirement of just in time compilation or instruction level emulation[14]. Various container technologies are Docker, Warden Container, Openvz, Framework. These allows the container engine to pack and run the applications as a container. [31][14]

2.4 Containerization vs Virtualization

Why container overcome the need of Virtual Machine ??

1. The Virtual Machines run on its own Virtual hardware and the OS kernel and the program associated to it are installed on the virtual hardware whereas the container shares the base OS which leads the container to be efficient enough in field of storage utilization[14].
2. The container takes relatively less time to boot than the VM.[14].
3. Containers can pack more applications in a single physical server as compared to the VM[37].
4. Containers boost up the performance and its efficiency as compared to VM because it eliminates the need of additional resources demand by individual OS[35].

Chapter 3

Virtual Machines Consolidation

With the rapid growth of Cloud Computing, to attract the cloud users, cloud providers are paying attention to provide high quality services meanwhile reducing the costs by reducing the energy consumption of the physical machines and to increase resource utilization in data centers. In recent years, many researches were done for data center energy-efficient techniques. Consolidation refers to efficiently use the computer resources and prevent the server as well as storage from getting under-utilized and more space consumption. Server Consolidation allows more than one servers to reside on Physical Server in order to Utilize all of a Server's available resources. To achieve better performance and enhances the service quality Server Consolidation is done over the Data centers. Virtual machine consolidation refers to consolidation numbers of VMs in a Single Server. Most of the research focused on Virtual machine consolidation technique and depending upon the modeling technique, problem solving technique were proposed.

Virtual machine consolidation is classified into:-

- Static VM consolidation
- Dynamic VM consolidation

3.1 Static VM consolidation

Static VM consolidation works with number of empty Physical machines and numbers of Virtual Machine with its resources requirements. Here, initially the VM placement is done basing on bin packing algorithm and are allocated to minimum numbers of active Physical machines to increase the resource utilization and reduce the energy consumption of the data center. [17] However, it cannot meet the SLAs requirement due to instability as well as unpredictable workload and dynamic placement of Virtual machine is not considered, which leads to SLAs violation. [38][26]

3.2 Dynamic VM consolidation

Dynamic consolidation of Virtual Machines in a Data Center is an effective way to reduce energy consumption and increase resource utilization. Here, VM placement algorithm aims at determining the most optimal Virtual Machine to Physical Machine mapping and Virtual Machine migration for placement re-optimization. In Dynamic VMC, Virtual Machines are consolidated considering dynamic workload and virtual machine location.

It focuses to relocate the Virtual Machine to minimum number of Physical Machine to reduce the active Physical Machine to minimize the energy cost. [26] Dynamic Virtual Machine consolidation is shown as an efficient method which improves the resource utilization in the data center. [20][24][21][10]

The complex problem of Dynamic Consolidation of Virtual Machine is splitted into four parts:

- **Host Overload Detection:** Determine when the host is overloaded and migrate the Virtual Machines from one host to another.

- **Host Under-load Detection:** Determine when the host is underload and migrate all the Virtual Machines from the host and switches the host to sleep mode.
- **VM Selection and Migration:** Selecting which Virtual Machine should be migrated from the host if the host is overloaded or underloaded.
- **VM Placement:** Finds a new host to place the selected Virtual Machines from the overloaded or the underloaded host.[6]

A threshold value is considered to prevent the CPU being 100% utilized which may lead to performance degradation of the Physical Machine.[16] To determine whether a host is overload,a static upper threshold of 85% for CPU utilization was proposed based on studying the workload.[20]Recently Dynamic CPU utilization was introduced.[6]To determine the underloaded host,a simple approached is proposed.The host which is minimum utilized select all its Virtual Machine and migrate it to other host and enable that host to sleep mode[6].

Chapter 4

Virtual Machines Placement

4.1 VM Selection Policy

Host consist of numbers of VMs,when a host's CPU utilization exceed the upper threshold,the next step is to migrate a VM from one host to another host.Now,to select the VM among the group of VM in the host we use Fuzzy VM selection technique.If randomly VMs are selected to migrate than the VMs which are efficiently using the resources might get migrated and the VMs which uses more resources and least efficient will run in the host which results in more energy consumption and least resource utilization in the data center.Therefore,in Fuzzy VM selection technique we take some input(i.e Minimum Migration Time,Maximum Correlation,Inference rules) where each input have some advantages over others and generate a Fuzzy value i.e which VM is selected for migration.Once a VM is selected to migrate,the host is checked again whether it is overloaded.If the host is overloaded again Fuzzy VM selection technique is used to migrate the VM until and unless the host is not overloaded.[6][18] Now focusing on the variables that are used as input in the Fuzzy selection technique are as follows:-

1. **Minimum migration Time** Minimum Migration time Policy selects the VM which requires minimum time to migrate from one host to another host as compared to other VMs present in

the host. The Migration time is calculated by the ratio between the amount of RAM utilized by the VM to the Network Bandwidth available for the host. Let $RAM(a)$ be the RAM utilized by the VM a and $RAM(b)$ be the RAM utilized by the VM b and V_v be the number of VMs present in the host. NET_v be the available network bandwidth in the host v.

$$a \in V_v \mid \forall b \in V_v, \frac{RAM(a)}{NET_v} \leq \frac{RAM(b)}{NET_v} \quad (4.1)$$

Minimum Migration time is considered as one of the input to the Fuzzy VM selection system.

2. **Maximum Correlation** This method works based on the idea that the higher the correlation between the resource usages by applications running on an oversubscribed server, the higher the probability of server being overloaded. It says higher the correlation between the CPU utilization of VM in a host, higher is the probability of the host to get overloaded. [39][4]

Correlation is considered as a predictive measure as it determine which VM is going to overload the host. So we will select those VM which have higher correlation between the CPU utilization of VMs. To find the correlation, we apply the *Multiple correlation coefficient*. It correspond to the squared correlation between the predicted and the actual values of the dependent variable. [2]

3. **Inference rule** Inference rule are generated using the variables (i.e RAM, Correlation). If RAM is less, more priority is given as migration is carried out as a faster pace. If Correlation is high, more the probability to overload the host, so higher the priority to migrate.

4.2 VM Placement Policy

The VM placement policy aims to select the most optimal Physical Machine to place the Selected Virtual machine. It focuses on the Virtual Machine to Physical Machine mapping whether it is the initial VM placement or the reallocation of the VM by VM migration.

The VM placement policy is basically classified into two type depending upon the goal of placement:

1. **Power-based approach:** Its goal is to get an VM-PM mapping technique which makes energy-efficient and most resource utilized system.[5]
2. **QoS-based approach:** Its goal is to get an VM-PM mapping technique which fulfill the Quality of service requirement.[10]

Different VM placement techniques are used by the authors based on linear integer program, Bin packing and genetic algorithm considering CPU, memory, bandwidth as resources. Their main goal is to reduce the number of Active Physical Machine, Reduce Migration Cost, Reduce energy consumption, Reduce network traffic and Increase the resource utilization.

Ghribi et al. [19] proposed Exact VM allocation algorithm and Exact VM migration algorithm based on Bin packing and linear integer program respectively. The objective of exact VM allocation algorithm is to pack VMs into set of servers or the hosting nodes based on their power consumption. The objective of exact VM migration algorithm is to migrate the VMs from the source nodes to the destination nodes. The combined algorithm aims to minimize the number of active servers or maximize the number of idle servers to sleep mode. It reduces the number of Active Physical Machine and decrease the migration cost. CPU is considered as resource to the Physical Machine. It provides better performance than Best-fit algorithm. The combined

algorithm is formulates as follows:-

$$\max M = \sum_{i=1}^{m'} P_{i,idle} y_i - \sum_{i=1}^{m'} \sum_{j=1}^{m'} \sum_{k=1}^{q_i} P'_k z_{ijk} \quad (4.2)$$

where $P_{i,idle}$ is the power consumed by the idle servers, y_i is equal to 1 if the server is idle i.e no VM is active in the server i and if one at least one VM is active in the server i. P_k is the cost defined of power consumed for migrating the VM and z_{ijk} is the migration of the VM k from server i to server j.

Bobroff et al. [10] introduced a new management algorithm based on Measuring historical data, Forecasting the future demand and Remapping VMs to PMs, which is referred as Measure-Forecast-Remap (MFR). The MFR is introduced for dynamic resource allocation, as it allows the live migration of VMs. The objective is to minimized the number of Active Physical Machine which are hosting the virtual machines where the rate of demand is restricted to overload the resource capacity and reduces the number of SLA violation. To minimize the number of physical machine, Time series forecasting technique and bin packing technique is used. The Virtual machines demand is forecast and depending upon the demand they are sorted in descending manner. Later, Each Vm is choose to place in the Physical Machine. It check the capacity of the PM and the allocation demand of the vm. If the demand is less than the capacity, it is placed in the PM else placed in a PM with higher capacity. CPU is considered as resource to the Physical Machine. It provides better performance than Static algorithm. The algorithm is formulated as:-

$$G(r) \approx 1 - \frac{E[U] + E_p(r)}{L_p} \quad (4.3)$$

where L_p is distribution of the demand probability density in term of p-percentile, $E_p(r)$ is the distribution of the predicted error in term of p-percentile and $E[U]$ is the mean demand distribution.

Wang et al. [40] formulate Group Packing algorithm which uses random variable to predict future demand of Bandwidth by the VMs based on Bin packing problem. The author captured VM bandwidth demand by random variables following probabilistic distributions. In Stochastic Bin Packing VMs with known bandwidth are consolidated onto servers such that the number of server used are minimum and the chance for violating the server size is below threshold. CPU and Bandwidth are used as Resource to the Physical Machine. The Stochastic Bin Packing Algorithm allows n VMs to be Consolidated onto a numbers of Physical servers. The VMs are identified with known bandwidth demands. The Author Consider a Probabilistic distribution instead of fixed values as considered in classical Bin Packing Problem. It provides better performance than First fit algorithm and First-fit Decreasing algorithm.

Chen et al. [12] present Effective VM sizing algorithm based on integer programming. It reduces the number of active Physical Machines. The main idea behind Effective VM sizing algorithm is, it takes VMs resource demand as a random variable and estimate aggregate resource demand of servers through intrinsic demand and correlation-aware demand. Resources considered are CPU demand and memory is taken as a constraint. It provides better performance than First-fit Decreasing algorithm and Harmonic algorithm. The algorithm is formulated as:-

$$ES_{ij} = ES_{ij}^I + ES_{ij}^{CA} \quad (4.4)$$

where ES_{ij} is the effective size of Vm i which is hosted on server j, ES_{ij}^I is the intrinsic demand of the effective size of Vm i which is hosted on server j and ES_{ij}^{CA} is the Correlation-aware demand of the effective size of Vm i which is hosted on server j.

Song et al. [36] present Variable item size bin packing problem(VISBP) where each PM is a bin and VM are the items which are packed into the PM.The VISBP algorithm is capable of overload avoidance and green computing.The resources of the data center are dynamically allocated during live Vm migration where the VMs migrate from one server to the other without hampering the applications running inside the VM.The algorithm was further extended as multi-dimentional VISBP which handles network intensive workloads in addition to CPU workload being considered earlier.It provides better performance than hot spot migration and load balance.

Singh et al. [34] proposed a load balancing algorithm called VectorDot based on multi-dimensional Knapsack.The author described their system Harmony that integrates storage virtualization and the servers which continuously track the resource usages of servers, storage nodes and network switches in the data center.The present state of the data center(virtualized) is the input to the load balancing algorithm.Based on the input to the algorithm the load balancer migrates the VM from one node to other and bring the overloaded node below the threshold value. CPU,memory,network are used as resource to the Physical Machine.It provides better performance than the Best-fit and First-fit algorithm.

Farahnakian et al. [15] present Linear Regression based CPU Utilization Prediction(LiRCUP).It is a load prediction method for VM consolidation in cloud data center.Using linear regression technique,the proposed method predict each host's CPU utilization.If predicted utilization value exceed current utilization capacity then the host is overloaded and VM are migrated to underutilized hosts. Moreover,the under utilized VM migrates all its VM to other host.CPU is used as the Resource to the Physical Machine.It provides better performance than the THR, MAD, IQR and LR on minimizing the energy cost and SLA violation rate.Based on the linear regression

method LiRCUP is used predict the approximate function. This function gives the current and the future CPU utilization of the host. The function is stated as follows:-

$$y = \beta_0 + \beta_1 x \quad (4.5)$$

where β_0 and β_1 are the parameter of regression co-efficient which estimate depending upon the last CPU utilization in the host.

Mi et al. [29] proposed genetic algorithm based online self-reconfiguration approach(GABA) which predicts the future workload of the applications and based on the predicted value, Genetic algorithm is used to find reconfiguration policy. It reduces the number of Active Physical Machine and increases the CPU utilization. CPU is the resource to the Physical Machine.

Chapter 5

Containers

Containers are light weight, which are easily managed and can be configured, which leads the Virtual Machines by decreasing start-up time. In containers, Application and data are bundled together in simpler form with greater performance oriented way than the Virtual Machines[27]. According to [8], Virtual machines lags behind the container in term of portability and the efficiency. Various container management systems are Docker, Linux Containers, Warden Container, OpenVZ.

A new Cloud service-Containers as a Service (CaaS) was introduced by Amazon Web Services and Google, to the Cloud Services-Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). Infrastructure as a Service provides virtualized compute resources and Platform as a Service provides Services and Language Runtime to developers. Containers as a Service (CaaS) is the layer which connects this layer together.

Key terms related to Containers are as follows:-

1. Chroot

It is a Linux command which changes the Root directory of current process and sub-ordinates to a new directory. Containers use it for isolation and to shares the file systems in the virtual environment.[14]

2. **cgroups**

It is a part of kernel which is used by the Container for Resource Management. cgroup limits the Resources i.e CPU, Processors, Memory etc [14]. cgroups manages the restriction of the resource utilization, moreover it prioritise the utilization of CPU and the memory [7]. The size of the container can be altered by limiting the cgroups. It creates isolated instances which inherits the value of the root file system. [3]

3. **namespaces**

At various levels, to create a barrier between the containers namespaces are used. Containers with its own process id is allowed by the **pid** namespaces. **net** namespace permits each containers to have its own iptables and routing tables. **IPC** namespaces isolates the semaphores and the shared memory portions. **mnt** namespaces allows the containers to have its own mountpoints [14]. As said by Biederman [9] kernel namespace is used to handle the container isolation.

5.1 Container Implementations

1. **Linux Containers**

These are the container which are light in weights and are supported by Linux i.e Ubuntu and Oracle. It is a OS-level virtualization cause it executes number of linux container on top of a single linux host.

A unique process id is assigned to each container and a single process run in each individual container. A private IP address and a private file system is assigned to each containers. Resource isolation is done with the help of cgroups and namespaces. Linux Virtual Machine can get the file system from the container which uniquely define the Linux Container. Apparmor Security is used by the containers to isolate the host and to check the device ac-

cess on the container it uses cgroups. It provides better isolation as compared to the chroot.[14]

Linux container have advantages over the Virtual machines and other container by its light weight and providing Filesystem as well as network isolation. Moreover, the non root user can also build container whereas it can not be availed in Docker[27]. To distribute the Linux container and to make easy utilization Flockport is used which ensures the software and the web stakes.

2. Docker

It is a platform which supports the container to deploy, ship and run distributed application. Docker follow an architecture with docker images, its container and the registry as its main components. It uses the Linux container for container implementation and add images into it. Docker uses Union file system to create the images[13].

Unique Process Id and a private Id is assigned to each container. The process id ensures that the processes in other container will not be affected by the processes running in other containers. Managing the network resources and isolating the resources of the system regarding the network is done with the help of net namespaces where as isolation of inter-process communication resources is done with ipc namespaces [27]. Resource isolation is done with the help of cgroups. Single process can run in each container. The transmission of packets between the host and the container is done with the help of Virtual Ethernet.[3] It is managed by a daemon tool as well as command line tool. Looking on the security issues, to gain advantages over the Linux container it undergoes Mapping of container root user is done to the docker non-root user and let the docker daemon to run as an non root user.

Docker utilizes only a single copy of the operating system which reduces the storage and utilizes less memory. It utilizes the im-

ages which is required by the containers with the help of Advanced Multi-Layered Unification File system.[14]

3. **Warden Container**

It can run on multiple Operating System hosting it. It is widely known as kernel independent implementation for containers. It is supported by Ubuntu and its design is quite familiar to the Operating System. It uses namespaces for process and network isolation where as Resource isolation is done by using cgroups. Unlike, Linux container multiple processes can run on a single container.[14]

4. **OpenVZ**

It uses dynamic realtime partitioning which manages virtual as well as physical servers. It uses a modified version of Linux Kernel with some additional extensions. Produce better performances than the Hypervisor. Each container is assigned with its process id and every container has its own memory and semaphores.

Resources Management is done with the help of CPU scheduler and Bean Counter. It uses net namespace with its own Virtual network device and IP address. Additionally, provides file isolation such as the system library and the application files. To regain the last state it provides checkpoints where recently stored data can be recovered in case of any failure. The complete process of a container can be stored in a file which enables the Live Migration of the containers [14].

5. **Google Imctfy**

It provides a resource configured API which overcomes the instability of Linux Containers API by simplifying the container management. Process isolation is done with the help of pid namespaces. Resource isolation is done with the help of cgroups and Network isolation is achieved by using net names-

paces where as File system isolation is achieved by the chroot.

6. **CoreOS**

CoreOS is designed in such a manner so that it can provide essential characteristic which are needed to operate the softwares. Rkt named as rocket container runtime is introduced by the CoreOS which helps in building a model for the container. It lead the Docker container implementation in the field of security issues.

Depending upon the characteristic of the servers, replica of virtual containers were introduced where deployment of the application was carried out automatically. It is Secured way of application deployment. Using Rocket is bit complex as compared to Docker as the process of creating a container with help of its interface is easier[14].

Table 5.1: Comparison of Containers

Containers	Resource	Process	Network	Host
Linux	cgroups and namespaces for isolation	unique process id	net namespaces for isolation	Kernel dependent, Apparmor security to isolate host
Docker	cgroups for isolation	ipc name-spaces for isolation	net name-spaces is used for isolation	Kernel dependent, uses single copy of OS
Warden	cgroups for isolation	Multiple process run on single container	net name-spaces for isolation	Kernel In-dependent, can run on multiple OS
OpenVZ	cpu scheduler, bean counter for Management	unique process id	own ip address and virtual network device	Kernel dependent
Google Imctfy	cgroups for isolation	pid name-spaces for isolation	net name-spaces for isolation	Kernel dependent
CoreOS	cgroups for isolation	unique process id	net namespaces for isolation	uses single copy of OS

5.2 Containers in action

1. Accelerating Big Data Applications

Containers are used to increase the speed of big data applications. Different applications vary in regard of their behavior and the requirement of resources. In cloud environment for big data, *Bhimani et al.*[8] compared numbers of virtualized frameworks and checked the resource utilization by Spark. Hadoop and Spark are some of the implementation of big data enterprise cloud environment such as MapReduce which helps to process the data in large scale and analyze it.[33]

As name suggested, MapReduce splits up in to map and reduce. "Map" dispatches the chunks of data from an large data set into individual system to process the data and "Reduce" gathers all the data from the map and produce an output. Meanwhile Spark ahead the Hadoop by overcoming its drawbacks i.e Disk Input Output Operation by its cache all the data into memory rather than storing it in disk. So, it does not require and disk access. Each container has shared memory and the page cache. For data analytics, Spark support numbers of applications such as library of machine learning and graph processing. It has one master and many workers nodes, where the master node broadcast the variables and creates accumulator and the workers node also termed as JVM, which runs numbers of executors and store data in the memory.

Looking upon implementation, Bridge Docker is used to connect different containers, a Docker file with DockerSparkImage is used in the docker framework to create instance of all containers. Firstly, ubuntu is instantiated and then on top of the ubuntu, hadoop spark and java is loaded and installed and committed. Produced a result on comparing different Virtualization on Spark, where the container has benefits of mapping, providing light weight and disk storage over the virtual machines.[8]

2. Deploy efficiently high performance computing applications

For performance evaluation, *Chung et al.*[13] used high performance computing tools likely Graph500 and High Performance Linpack (HPL). For measuring computing ability a dense linear system is solved with the help of HPL and Graph500 which is a graph algorithm used for data intensive applications. Basically, computing ability and data traceability is considered for performance evaluation.

Docker is a platform which support containers and applications are deployed which shares the libraries and the dependencies in an host system. So, it overcomes the overhead problem faced in the virtual machines. For deploying application, the docker shares the same os kernel along with the libraries. For testing, with increasing number of instances keeping track on CPU, execution process and RAM was performed.

For executing an application through HPL it uses HPL package, math libraries and a message passing interface (MPI). Math library named OpenBLAS and message passing interface named OpenMPI are installed in the host. Within a host OS, its libraries which are needed for HPL to run a process are embedded inside the container. Graph500 is used to represent data intensive applications which return in 2 part i.e undirected graph construction and Breadth-First Search and produced an output TEPS (traversed edges per second). So container excell the Virtual Machines in computing ability and data traceability.

3. Increasing The Diversity of Resilient Server

With adopting Virtualization many benefits were availed but at the mean time threats for computer servers increases drastically. So to check the threats and to make an resilient server, *Winarno et al.* [41], using Technology of Linux container (LXC) created a self repairing network (SRN). For a Virtual Machine Manager-based SRN, Xen was used as the VMM where the server was resilient to particular threats and was found with many holes in the virtualized environment such as the service denial, communication between the host and the Virtual Machines. However, the network cleaning problem [23] was solved when SRN is proposed. Basic model of SRN is categorized into mutual repair and self-repair. The mutual repair, repairs the other nodes where as the self-repair, repairs itself. Here, the two method is combined and names as mixed-repair. Focusing on the resilient server, container proved to be better than the virtual machines as it recovers the abnormal node much faster than the virtual machine manager based SRN and consumes less memory, meanwhile resisting the growing threats to the server.

5.3 Container PLacement

Nardelli et al. proposed as Elastic provisioning of Virtual machines for Containers Deployment (EVCD) based on Integer linear programming problem considering computing resources and different containers requirements. It focuses to solve the deployment problems in the container i.e to identify the elastic set of the host executing the containers. Greedy first-fit and round-robin were considered as the heuristic to solve the deployment problem of container and check weather improvement of Quality of Service can be achieved.

Basically, EVCD is used to determine the mapping of the virtual machines onto the containers[30]. It is formulated as follows:-

$$F(x, z) = w_d \cdot \frac{D(x, z) - D_{min}}{D_{max} - D_{min}} + w_c \cdot \frac{C(z) - C_{min}}{C_{max} - C_{min}} \quad (5.1)$$

where w_d and w_c define the attributes of QoS, D_{max} and D_{min} are maximum and minimum value of the expected time of deployment, C_{max} and C_{min} are maximum and minimum value of the expected cost of deployment.

Casalicchio et al. produced an solution to Elastically solve Provisioning of Virtual Machines for Container Deployment problems. He focused on the scaling properties of management of container over distributed data centers. In repositories, containers are stored. The repository is named as Container Image Registries. The initiation of containers execution as well as the selection of the node where the container will execute is decided by the orchestrator. The nodes together combined to form a cluster. Numbers of containers can run inside a node, where the Container Engine(CE) act as an manager which manages the executions of the container. Each container engine communicates with the orchestrator for the execution of the container. Apart from the orchestration problem it reduces the overhead by enhancing the management of the network and secure the resource sharing i.e isolating for enabling the multi-tenancy[11].

Piraghaj et al. proposed a framework which consolidate the container into the virtual machines. The main focus is to improve the efficiency of the server in term of energy. It is modeled to achieve Container as a service environment which lies in between Infrastructure as a service and Platform as a service. In CaaS model applications are run inside the containers, where numbers of container are deployed inside the Virtual Machine which are hosted on top of the physical server. The main idea is to consolidate the container into minimum number of VM and VMs into minimum number of servers.

It is classified into host status module and consolidation module. In host status module, it detects whether the host is overloaded or underloaded, and the container selector selects the container and the selected containers are stored in the container migration list and are submitted to the consolidation module. In the consolidation module, in overloaded host list stores the detail of the host which are overloaded, overloaded destination selector selects the destination to place the container from the overloaded host and in underloaded host list stores the detail of the host which are underloaded, underloaded destination selector selects all the containers to place them into a vm. To minimize the power consumption, it is formulated as

$$P_{dc}(t) = \sum_{i=1}^{N_S} P_i(t) \quad (1)$$

where $P_{dc}(t)$ is the data center power consumption at time t , $P_i(t)$ denotes server i power consumption and N_S is the number of servers.

Table 5.2: Comparisons of Algorithm and Techniques in Virtualization

Author	Algorithm/Technique	Resources	Achievements
Ghribi et al	Exact Vm allocation and exact vm migration based on bin packing and linear integer program.	CPU	Minimize active physical machine and maximize idle physical machine
Bobroff et al.	Measure-Forecast-Remap (MFR) algorithm uses Time series forecasting technique and bin packing technique	CPU	Minimize active physical machine
Wang et al	Group Packing algorithm based on Stochastic bin packing problem	CPU, Bandwidth	Minimize active physical machine
Chen et al	Effective VM sizing algorithm based on integer programming.	CPU	Minimize active physical machine
Song et al	Variable item size bin packing problem(VISBP) based on bin packing.	CPU, Network, Memory	Balancing the load
Singh et al	VectorDot based on multi-dimensional Knapsack	CPU, Network, Bandwidth, Memory	Manage the overloaded load and dynamically balance the load
Farahnakian et al	Linear Regression based CPU Utilization Prediction (LiRCUP) based on linear regression.	CPU	Predict the current and the future CPU utilization of the host
Mi et al	Genetic algorithm based online self-reconfiguration approach(GABA).	CPU	Minimize active physical machine and Predicts the future workload of the application
Ferdaus et al	Static VM Placement based on bin packing.	CPU, Memory, Network	Increase Resource utilization and reduce energy Consumption.

Table 5.3: Comparisons of Algorithm and Techniques in Containerization

Author	Algorithm/Technique	Resources	Achievements
Nardelli et al	Elastic provisioning of Virtual machines for Containers Deployment (EVCD) based on Integer linear programming problem	CPU	Identify the elastic set of the host executing the containers
Casalicchio et al	On scaling management properties of container over distributed data centers	CPU, Memory	Enhance the management of network and secure the resource sharing
Piraghaj et al	Framework which consolidate the container into the virtual machines.	CPU, Network, Memory	Improved the efficiency of the server in term of energy

Chapter 6

Blockchain based Container Migration

6.1 Blockchain

A Blockchain is a decentralized distributed infrastructure, public ledger which stores the transactions for the crypto-currencies. The word Blockchain is termed as collections of complete blocks which are linked to each others in a chain network structure. The first block is known as Genesis block. The Blockchain have the complete detail of transaction (for crypto-currencies) from the Genesis Block to the most recent complete block.

Each block consist of block index, a timestamp, data, previous block hash value and present block generated hash value.

In Blockchain, once a transaction is stored in the block it cannot be altered or manipulated. Block can be added into the Blockchain with the help of Cryptography. The cryptography function is based on the executing the block details in an Merkle-tree structure. Basically, SHA-256 Algorithm is used for hashing in a block.

Table 6.1: Blockchain importance on Container migration

Blockchain Resources	Why Blockchain for Container ?
Blockchain uses financial transactions as data in its Blocks	Containers migration List can be used as data in the Block
Blockchain's data is available in numbers of systems in real-time	Containers migration data will be available in numbers of data center
It exists only in a peer to peer network	It will exists in core P2P n/w of Virtualized Data centers
Cryptography is used to prove the identity and the authenticity	Cryptography is used to uniquely identify the container to migrate and to authenticate
There is a need of ledger containing transactions details in Blocks	There is a need of ledger containing container migration details in Blocks
Difficult to change the data once transaction occur	Once the container migration occur, it is difficult to manipulate the list

6.2 Blockchain Working Model

When a new block is added into a Blockchain, before the block is added, all the nodes in the Blockchain execute the cryptography algorithm and depending upon the consensus protocol, if majority of the nodes or data miner agree for the authenticity of the transaction then the transaction is added into the Blockchain. After the transaction is added into the Blockchain and all other nodes get updated of the added block in their ledger. Lastly, the transaction is completed.

6.3 Literature survey for Blockchain based migration

Piraghaj et al.[32] proposed a system model where Containers run on top of Vm and application runs inside the containers. For Container consolidation, if a data center gets overloaded or underloaded, containers are migrated based on their selection policy and placement policy. The containers to be migrated are stored in *Container migration list*. *Alvarenga et al.*[1] proposed a block-chain based secured management and migration on Virtual network function. Apart from securing the accessibility of the data in blockchain, the transaction between the peer is ensured by a consensus protocol before it gets stored into the blocks. So attacks are not possible as it is ensured by the peers.

6.4 Proposed Model

Based on the survey, keeping container migration into focus, we extended the Blockchain technology into the container migration to prevent the malicious access of the container migration list. So we can prevent the container from being traced by the intruder who always tries to access the services running inside the containers. If the migration list is accessed, the intruder can alter the migration list value, for which the placement won't take place. As a result, if a host is detected as overloaded, it will remain in an overloading state, and if a host is underloaded, it will remain in an underloaded state. So, if the container migration list is encapsulated as block data inside a Block in a Blockchain, it turns too difficult for an intruder to access the data (i.e. container migration) from a block, meanwhile smooth execution of container migration is done. We proposed a model where the container migration list is stored inside a Block in a Blockchain.

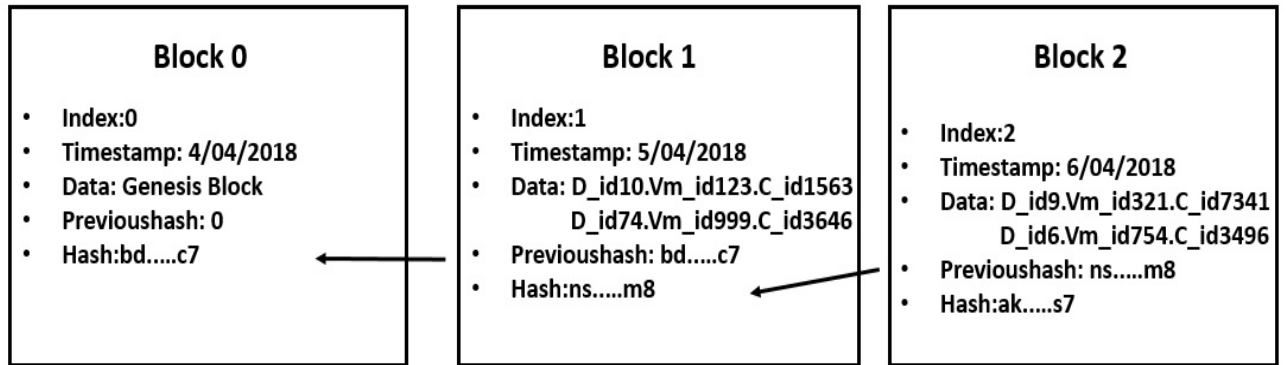


Figure 6.1: Proposed model

6.5 Implementation

We have used Node.js as an open source, cross-platform runtime environment for simulation supporting our research. Node.js is an event-driven application which provides a non-blocking I/O API and subsequently optimizes an application's throughput and scalability. Hence Node.js is suitable for server-side and networking applications.

Block structure:-

A block structure is created including the necessary details of a block like index, timestamp, data, previous hash and hash.

Listing 6.1: Block Structure

```

class block{
    constructor(index , timestamp , data , previoushash = ''){
        this.index=index;
        this.timestamp=timestamp;
        this.data=data;
        this.previoushash=previoushash;
        this.hash = this.calhash();
    }
}
  
```

Block hash:-

The block is hashed so that integrity of the data is achieved. SHA-256 algorithm is used for hashing which generate the hash of the block. Hashing is done with index, previous block hash value, timestamp and the data of the block as the inputs to the hashing algorithm. So, Import the SHA256 algorithm from node.js command prompt

```
npm install --save crypto-js
```

and include the SHA256 algorithm before defining the hashing algorithm and add the function *calculatehash()* inside the block class to generate the 64 bit hash value for the block.

Listing 6.2: Block Hashing

```
calchash () {  
  return SHA256(this.index + this.previoushash +  
    this.timestamp + JSON.stringify(this.data)).toString();  
}
```

Create Genesis Block:-

The Genesis block is the first block in the Blockchain. It consists of the index value, the timestamp when the block is created, the data i.e. the container migration list and the hash value of the block. It won't have the previous hash value of the block as because it is the initial block in the Blockchain.

Listing 6.3: Genesis Block

```
class blockchain {  
  constructor () {  
    this.chain = [this.genesisblock ()];  
  }  
  
  genesisblock () {  
    return new block (0, "5/04/2018",
```

```

        "Genesis Block", "0");
    }
}

```

now,an object was created of the class blockchain and reference to the created object was declared.

```

let souravchain = new blockchain();

```

Adding Block to the Blockchain:-

To add a block we must know the hash value of the previous block.Initially a block is created in the blockchain with its index value,timestamp and the data.

Listing 6.4: Adding Block with Container Migration list

```

souravchain.addblock(new block(1,"6/04/18",
                                "Dc_id11.Vm_id314.C_id5454,
                                Dc_id31.Vm_id574.C_id2694"));

```

then the previous hash value is fetched and stored in the block.

Listing 6.5: Fetch Previous Block

```

getrecentblock(){
    return this.chain[this.chain.length -1];
}

```

later the hash value of the current block is calculated with the index value,timestamp,data,the previous block hash value.

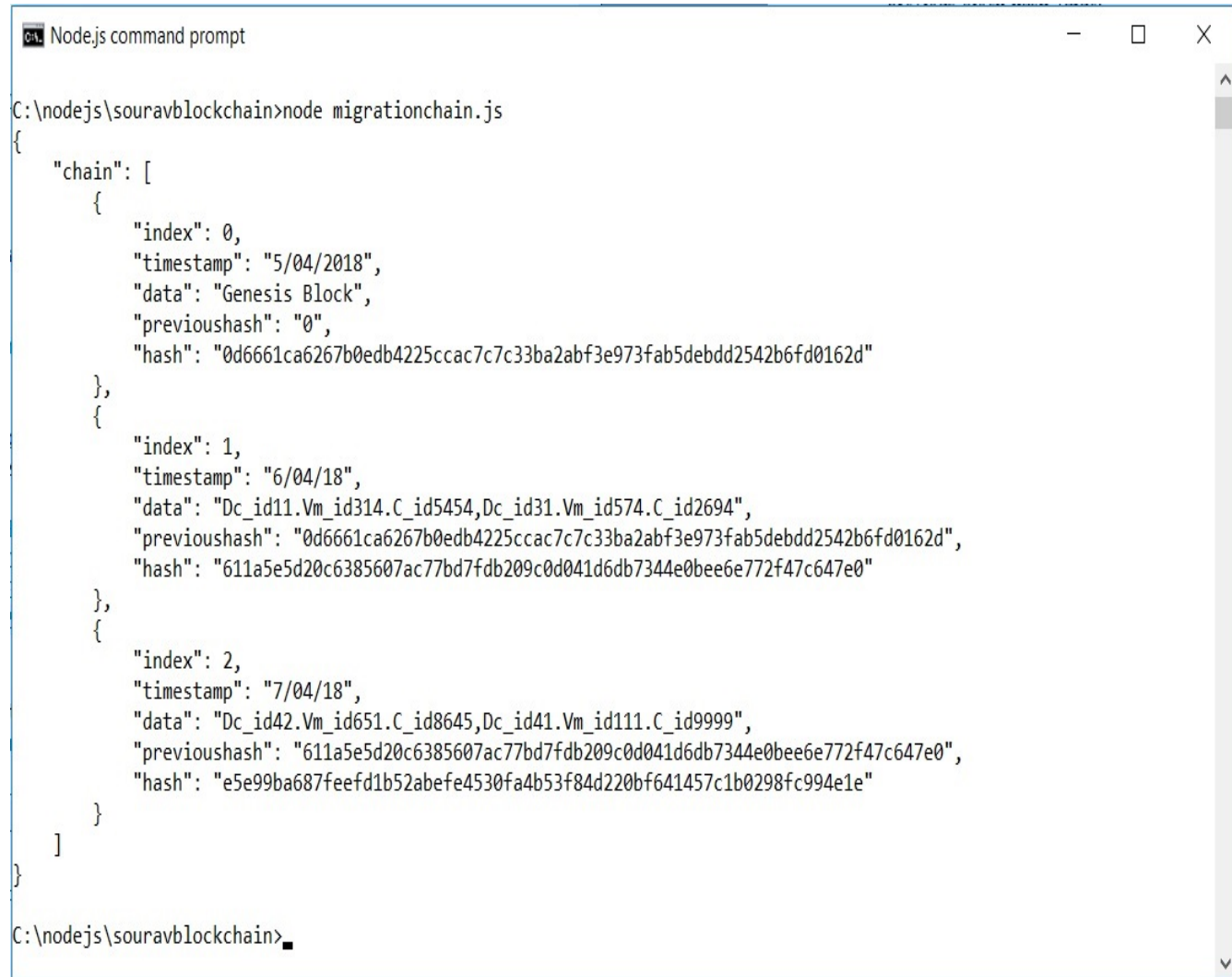
Listing 6.6: Calculating hash value and add block

```

addblock(newblock){
    newblock.previoushash=this.getrecentblock().hash;
    newblock.hash= newblock.calchash();
    this.chain.push(newblock);
}

```

Now the Blocks were created in the blockchain with the genesis block as the first block and each block keeping record of the data and previous block hash value.



```
Node.js command prompt
C:\nodejs\souravblockchain>node migrationchain.js
{
  "chain": [
    {
      "index": 0,
      "timestamp": "5/04/2018",
      "data": "Genesis Block",
      "previoushash": "0",
      "hash": "0d6661ca6267b0edb4225ccac7c7c33ba2abf3e973fab5debdd2542b6fd0162d"
    },
    {
      "index": 1,
      "timestamp": "6/04/18",
      "data": "Dc_id11.Vm_id314.C_id5454,Dc_id31.Vm_id574.C_id2694",
      "previoushash": "0d6661ca6267b0edb4225ccac7c7c33ba2abf3e973fab5debdd2542b6fd0162d",
      "hash": "611a5e5d20c6385607ac77bd7fdb209c0d041d6db7344e0bee6e772f47c647e0"
    },
    {
      "index": 2,
      "timestamp": "7/04/18",
      "data": "Dc_id42.Vm_id651.C_id8645,Dc_id41.Vm_id111.C_id9999",
      "previoushash": "611a5e5d20c6385607ac77bd7fdb209c0d041d6db7344e0bee6e772f47c647e0",
      "hash": "e5e99ba687feefd1b52abefe4530fa4b53f84d220bf641457c1b0298fc994e1e"
    }
  ]
}
```

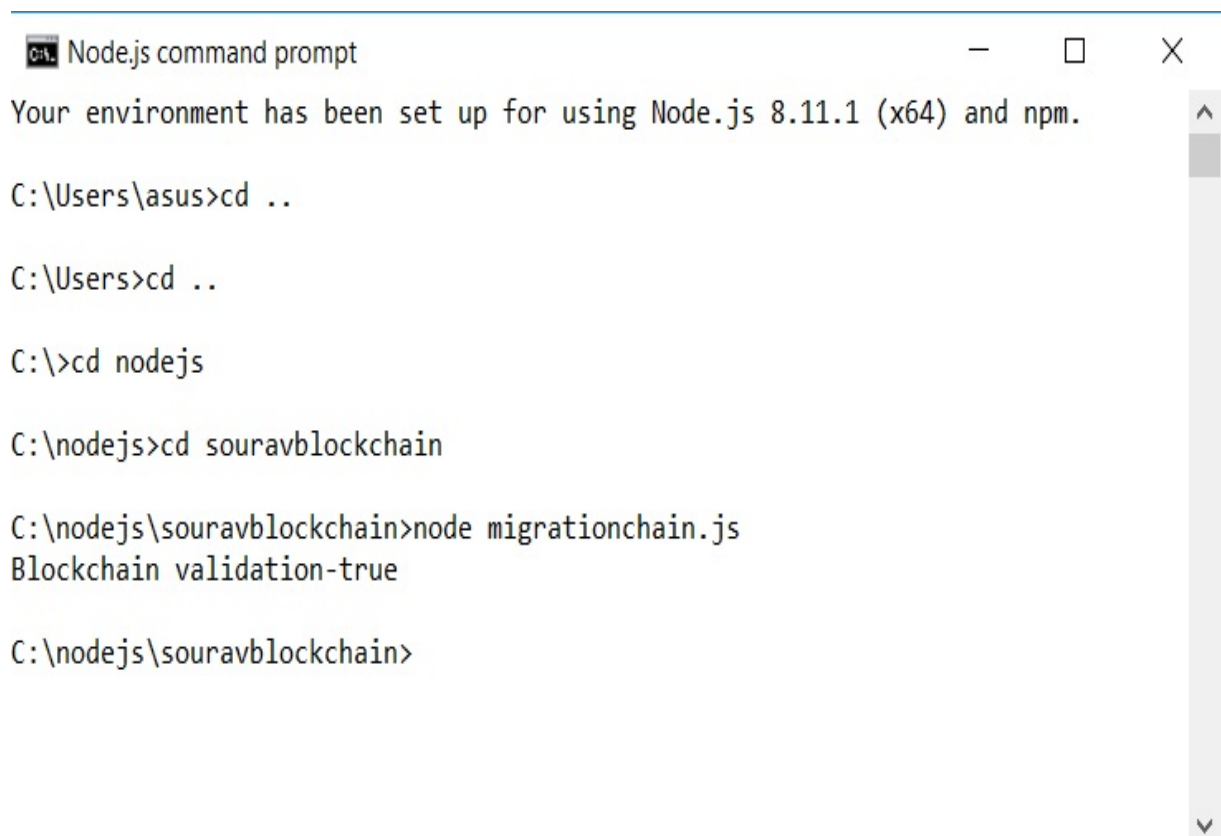
C:\nodejs\souravblockchain>■

Figure 6.2: Blockchain with linked Blocks

Validating the Block:-

Once the transaction is stored in the block, it cannot be altered because if the data once stored and if any malicious access makes changes it violates the security aspect of the blockchain and for its distributed ledger all the nodes or the data miner will access the wrong data value.

To check validation, same data is implicitly changed by updating the data with block index, but if inputs are same to the SHA-256 algorithm it produces the same 64 bit hash value, so on comparing it shows that the Blockchain validation is successfully completed.



```
Node.js command prompt
Your environment has been set up for using Node.js 8.11.1 (x64) and npm.

C:\Users\asus>cd ..

C:\Users>cd ..

C:\>cd nodejs

C:\nodejs>cd souravblockchain

C:\nodejs\souravblockchain>node migrationchain.js
Blockchain validation-true

C:\nodejs\souravblockchain>
```

Figure 6.3: Valid Blockchain

To alter the existing block data, with new data value it is initialized by a different value.

Listing 6.7: Implicitly changing the block data

```
souravchain.chain[1].data="Dc_id10.Vm_id103.C_id1007 ,  
Dc_id9.Vm_id111.C_id3406";
```

If data is altered, a new hash will be calculated cause the block hash is calculated from its index, timestamp, data value and previous hash value.

Listing 6.8: Re-calculation of block hash

```
souravchain.chain[1].hash=souravchain.chain[1]  
.calchash();
```

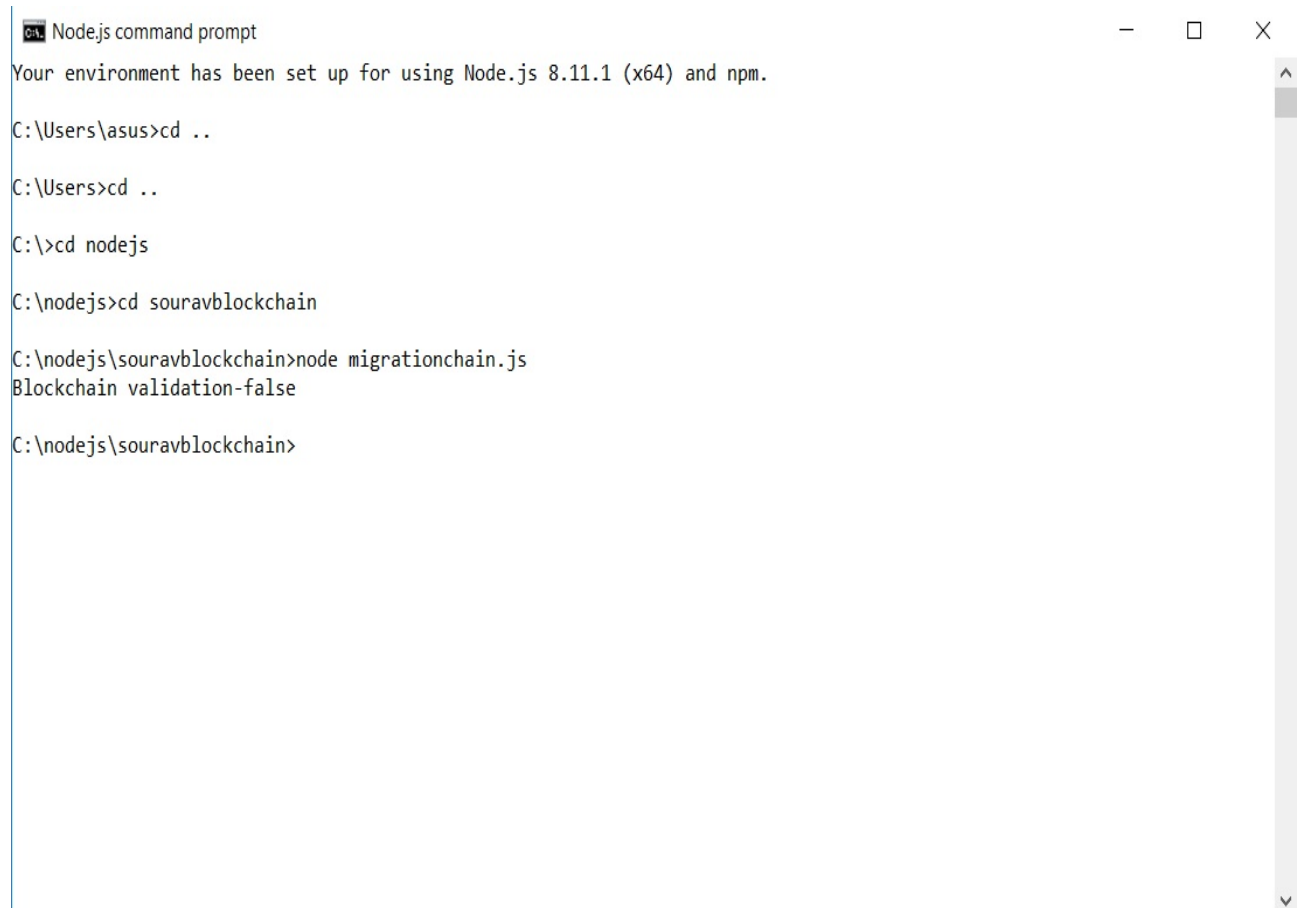
Listing 6.9: Validation function

```
ischainvalid(){  
  for(let i=1;i < this.chain.length;i++){  
    const currentBlock=this.chain[i];  
    const previousBlock=this.chain[i - 1];  
    if(currentBlock.hash !== currentBlock.calchash()){  
      return false;  
    }  
  
    if(currentBlock.previoushash !== previousBlock.hash){  
      return false;  
    }  
  }  
  return true;  
}
```

Check weather the block data is validates:-

Listing 6.10: Check Validation

```
console.log('Blockchain validation -' +  
souravchain.ischainvalid());
```



```
Node.js command prompt
Your environment has been set up for using Node.js 8.11.1 (x64) and npm.

C:\Users\asus>cd ..

C:\Users>cd ..

C:\>cd nodejs

C:\nodejs>cd souravblockchain

C:\nodejs\souravblockchain>node migrationchain.js
Blockchain validation-false

C:\nodejs\souravblockchain>
```

Figure 6.4: Invalid Blockchain

Chapter 7

Conclusion

The intruders with malicious intents more often trace the application running in a container, subsequently succeed in attacking the services. Certainly, the Blockchain based approach prevents the traceability of containers across the Virtualized Data centers networks which leads to robust and secured migration process.

Bibliography

- [1] Igor D Alvarenga, Gabriel AF Rebello, and Otto Carlos MB Duarte, *Securing configuration management and migration of virtual network functions using blockchain*.
- [2] Anju Bala and Inderveer Chana, *Vm migration approach for autonomic fault tolerance in cloud computing*, Proceedings of the International Conference on Grid Computing and Applications (GCA), The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2013, p. 18.
- [3] Rabindra K Barik, Rakesh K Lenka, K Rahul Rao, and Devam Ghose, *Performance analysis of virtual machines and containers in cloud computing*, Computing, Communication and Automation (ICCCA), 2016 International Conference on, IEEE, 2016, pp. 1204–1210.
- [4] Anton Beloglazov, *Energy-efficient management of virtual machines in data centers for cloud computing*, University of Melbourne, Department of Computing and Information Systems, 2013.
- [5] Anton Beloglazov and Rajkumar Buyya, *Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers.*, MGC@ Middleware, 2010, p. 4.
- [6] Anton Beloglazov and Rajkumar Buyya, *Optimal online deterministic algorithms and adaptive heuristics for energy and per-*

- formance efficient dynamic consolidation of virtual machines in cloud data centers*, Concurrency and Computation: Practice and Experience **24** (2012), no. 13, 1397–1420.
- [7] David Bernstein, *Containers and cloud: From lxc to docker to kubernetes*, IEEE Cloud Computing **1** (2014), no. 3, 81–84.
 - [8] Janki Bhimani, Zhengyu Yang, Miriam Leeser, and Ningfang Mi, *Accelerating big data applications using lightweight virtualization framework on enterprise cloud*, High Performance Extreme Computing Conference (HPEC), 2017 IEEE, IEEE, 2017, pp. 1–7.
 - [9] Eric W Biederman and Linux Networx, *Multiple instances of the global linux namespaces*, Proceedings of the Linux Symposium, vol. 1, Citeseer, 2006, pp. 101–112.
 - [10] Norman Bobroff, Andrzej Kochut, and Kirk Beaty, *Dynamic placement of virtual machines for managing sla violations*, Integrated Network Management, 2007. IM’07. 10th IFIP/IEEE International Symposium on, IEEE, 2007, pp. 119–128.
 - [11] Emiliano Casalicchio, *Autonomic orchestration of containers: Problem definition and research challenges*, 10th EAI International Conference on Performance Evaluation Methodologies and Tools. EAI, 2016.
 - [12] Ming Chen, Hui Zhang, Ya-Yunn Su, Xiaorui Wang, Guofei Jiang, and Kenji Yoshihira, *Effective vm sizing in virtualized data centers*, Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on, IEEE, 2011, pp. 594–601.
 - [13] Minh Thanh Chung, Nguyen Quang-Hung, Manh-Thin Nguyen, and Nam Thoai, *Using docker in high performance computing applications*, Communications and Electronics (ICCE), 2016 IEEE Sixth International Conference on, IEEE, 2016, pp. 52–57.

- [14] Rajdeep Dua, A Reddy Raja, and Dharmesh Kakadia, *Virtualization vs containerization to support paas*, Cloud Engineering (IC2E), 2014 IEEE International Conference on, IEEE, 2014, pp. 610–614.
- [15] Fahimeh Farahnakian, Pasi Liljeberg, and Juha Plosila, *Lircup: Linear regression based cpu usage prediction algorithm for live migration of virtual machines in data centers*, Software Engineering and Advanced Applications (SEAA), 2013 39th EUROMICRO Conference on, IEEE, 2013, pp. 357–364.
- [16] Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, Juha Plosila, Nguyen Trung Hieu, and Hannu Tenhunen, *Energy-aware vm consolidation in cloud data centers using utilization prediction model*, IEEE Transactions on Cloud Computing (2016).
- [17] Md Hasanul Ferdaus and Manzur Murshed, *Energy-aware virtual machine consolidation in iaas cloud computing*, Cloud Computing, Springer, 2014, pp. 179–208.
- [18] Xiong Fu and Chen Zhou, *Virtual machine selection and placement for dynamic consolidation in cloud computing environment.*, Frontiers of Computer Science **9** (2015), no. 2, 322–330.
- [19] Chaima Ghribi, Makhlouf Hadji, and Djamal Zeghlache, *Energy efficient vm scheduling for cloud data centers: Exact allocation and migration algorithms*, Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on, IEEE, 2013, pp. 671–678.
- [20] Daniel Gmach, Jerry Rolia, Ludmila Cherkasova, Guillaume Belrose, Tom Turicchi, and Alfons Kemper, *An integrated approach to resource pool management: Policies, efficiency and quality metrics*, Dependable Systems and Networks With FTCS

and DCC, 2008. DSN 2008. IEEE International Conference on, IEEE, 2008, pp. 326–335.

- [21] Brian Guenter, Navendu Jain, and Charles Williams, *Managing cost, performance, and reliability tradeoffs for energy-aware server provisioning*, INFOCOM, 2011 Proceedings IEEE, IEEE, 2011, pp. 1332–1340.
- [22] Keiko Hashizume, David G Rosado, Eduardo Fernández-Medina, and Eduardo B Fernandez, *An analysis of security issues for cloud computing*, Journal of Internet Services and Applications **4** (2013), no. 1, 5.
- [23] Yoshiteru Ishida and Kei-ichi Tanabe, *Dynamics of self-repairing networks: transient state analysis on several repair types*, International Journal of Innovative Computing, Information and Control **10** (2014), no. 1.
- [24] Gueyoung Jung, Matti A Hiltunen, Kaustubh R Joshi, Richard D Schlichting, and Calton Pu, *Mistral: Dynamically managing power, performance, and adaptation cost in cloud infrastructures*, Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on, IEEE, 2010, pp. 62–73.
- [25] Dong-Ki Kang, Fawaz Alhazemi, Seong-Hwan Kim, and Chan-Hyun Youn, *Dynamic virtual machine consolidation for energy efficient cloud data centers*, International Conference on Cloud Computing, Springer, 2015, pp. 70–80.
- [26] Md Anit Khan, Andrew Paplinski, Abdul Malik Khan, Manzur Murshed, and Rajkumar Buyya, *Dynamic virtual machine consolidation algorithms for energy-efficient cloud resource management: A review*, Sustainable Cloud and Energy Services, Springer, 2018, pp. 135–165.

- [27] Zhanibek Kozhimbayev and Richard O Sinnott, *A performance comparison of container-based technologies for the cloud*, Future Generation Computer Systems **68** (2017), 175–182.
- [28] Peter Mell and Tim Grance, *The nist definition of cloud computing*, 2010.
- [29] Haibo Mi, Huaimin Wang, Gang Yin, Yangfan Zhou, Dianxi Shi, and Lin Yuan, *Online self-reconfiguration with performance guarantee for energy-efficient large-scale cloud computing data centers*, Services Computing (SCC), 2010 IEEE International Conference on, IEEE, 2010, pp. 514–521.
- [30] Matteo Nardelli, Christoph Hochreiner, and Stefan Schulte, *Elastic provisioning of virtual machines for container deployment*, Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering Companion, ACM, 2017, pp. 5–10.
- [31] Claus Pahl, *Containerization and the paas cloud*, IEEE Cloud Computing **2** (2015), no. 3, 24–31.
- [32] Sareh Fotuhi Piraghaj, Amir Vahid Dastjerdi, Rodrigo N Calheiros, and Rajkumar Buyya, *A framework and algorithm for energy efficient container consolidation in cloud data centers*, Data Science and Data Intensive Systems (DSDIS), 2015 IEEE International Conference on, IEEE, 2015, pp. 368–375.
- [33] Abdul Ghaffar Shoro and Tariq Rahim Soomro, *Big data analysis: Apache spark perspective*, Global Journal of Computer Science and Technology (2015).
- [34] Aameek Singh, Madhukar Korupolu, and Dushmanta Mohapatra, *Server-storage virtualization: integration and load balancing in data centers*, Proceedings of the 2008 ACM/IEEE conference on Supercomputing, IEEE Press, 2008, p. 53.

- [35] Sachchidanand Singh and Nirmala Singh, *Containers & docker: Emerging roles & future of cloud technology*, Applied and Theoretical Computing and Communication Technology (iCATccT), 2016 2nd International Conference on, IEEE, 2016, pp. 804–807.
- [36] Weijia Song, Zhen Xiao, Qi Chen, and Haipeng Luo, *Adaptive resource provisioning for the cloud using online bin packing*, IEEE Transactions on Computers **63** (2014), no. 11, 2647–2660.
- [37] SJ Vaughan-Nichols, *Containers vs. virtual machines: How to tell which is the right choice for your enterprise*, 2015.
- [38] Akshat Verma, Puneet Ahuja, and Anindya Neogi, *pmapper: power and migration cost aware application placement in virtualized systems*, Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware, Springer-Verlag New York, Inc., 2008, pp. 243–264.
- [39] Akshat Verma, Gargi Dasgupta, Tapan Kumar Nayak, Pradipta De, and Ravi Kothari, *Server workload analysis for power minimization using consolidation*, Proceedings of the 2009 conference on USENIX Annual technical conference, USENIX Association, 2009, pp. 28–28.
- [40] Meng Wang, Xiaoqiao Meng, and Li Zhang, *Consolidating virtual machines with dynamic bandwidth demand in data centers*, INFOCOM, 2011 Proceedings IEEE, IEEE, 2011, pp. 71–75.
- [41] I Winarno, T Okamoto, Y Hata, and Y Ishida, *Implementing srn for resilient server on the virtual environment using container*, Intelligent systems Research Progress Workshop, 2015.