Capacity Planning System

January 18, 2018

Authors

KEERTHIVAASAN K

MURALIDHARAN R

NATARAJAN M

Guide

KUMARAN T

Assistant Professor

Dept. of CSE, SVCE

Table of Contents

List of Figures

1.0. Introduction

1.1. Purpose

The purpose of this document is to present a detailed description of the Capacity Planning System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do during starvation condition, the constraints under which it must operate and how the system will react to various situation in the Cloud Environment.

1.2. Scope of Project

This software system will be a Capacity Planning System for balancing the load of physical servers in a Virtualized Cloud Environment. This system will be designed to minimize the Load in the Physical Machine by providing suitable Algorithm to assist in automating the Selection and Migration process of VMs, which would otherwise have to be performed manually. By minimizing the Load in the Physical Machine it will not starve for Resources.

More specifically, this system is designed to monitor the Cloud environment with a set of constraints and threshold , so that it won't end up in starvation of physical machine. The system will also enable the live migration of the VMs in order to reduce the service Down time.

1.3. Glossary

Term	Definition
Cloud	It is the entire architecture in which the virtual environment can be deployed to provide services.
Cluster	Group of data centers
Datacenter	Collection of all the physical machine or host and monitors these host.
Downtime	The time for which the service is not available is known as service downtime.
Host	The physical server or machine in which the VMs are

	stored.
Mips	Million Instructions per Second
Network Bandwidth	The speed at which the data in the network transfers.
Pe	Processing environment
VM	Abbreviated as Virtual Machine, which is an emulation of a computer system.
User	Reviewer or Author.

1.4. Overview of Document

The next chapter, the Overall Description section, of this document gives an overview of the functionality of the product. It describes the system environment and is used to establish a context of the system in the next chapter.

2.0. Overall Description

2.1 System Environment

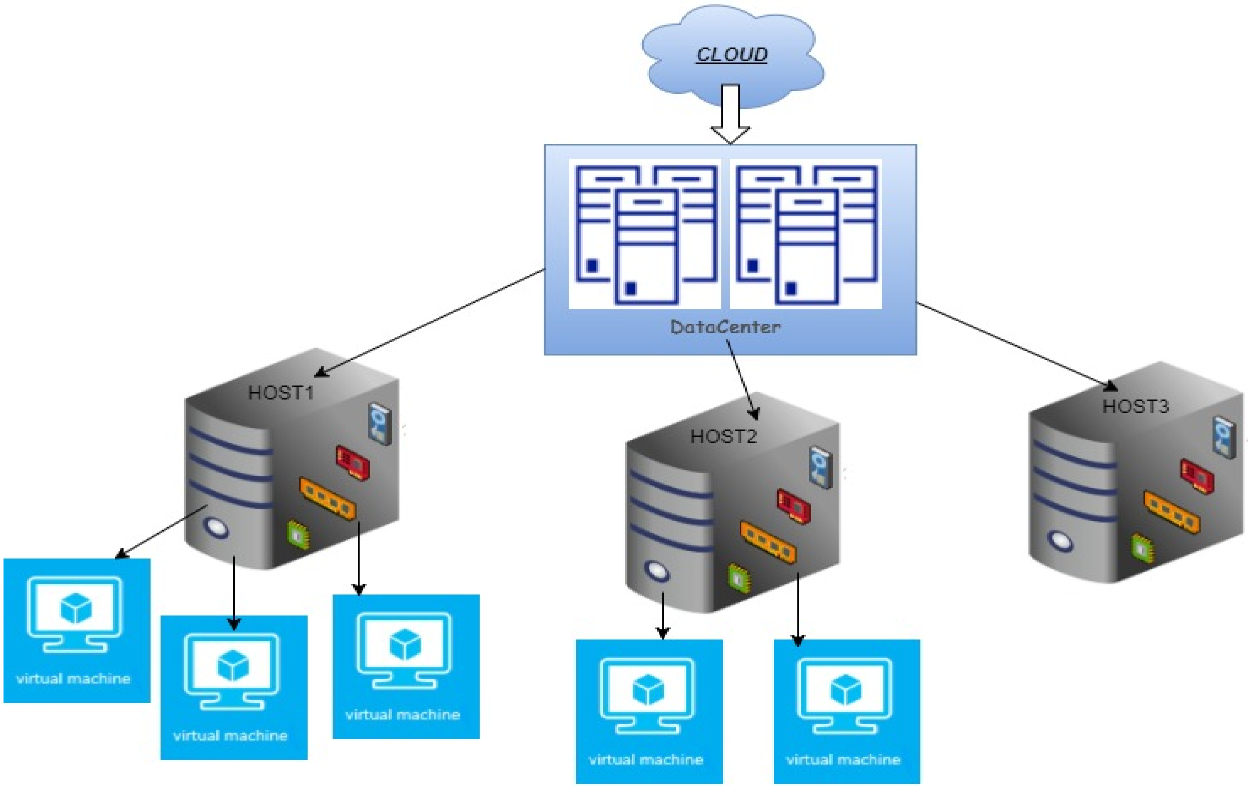


Figure 1 - System Environment

The System will have a simpler cloud environment with one Datacenter which itself contains the Physical machine. Each is called Host which will run the VMs . The VMs are the emulation of the computer systems that are virtually present in the cloud .

2.2 System overview

This section outlines the overall working of the system, this includes the various activity like VM allocation, Starvation check and Migration.

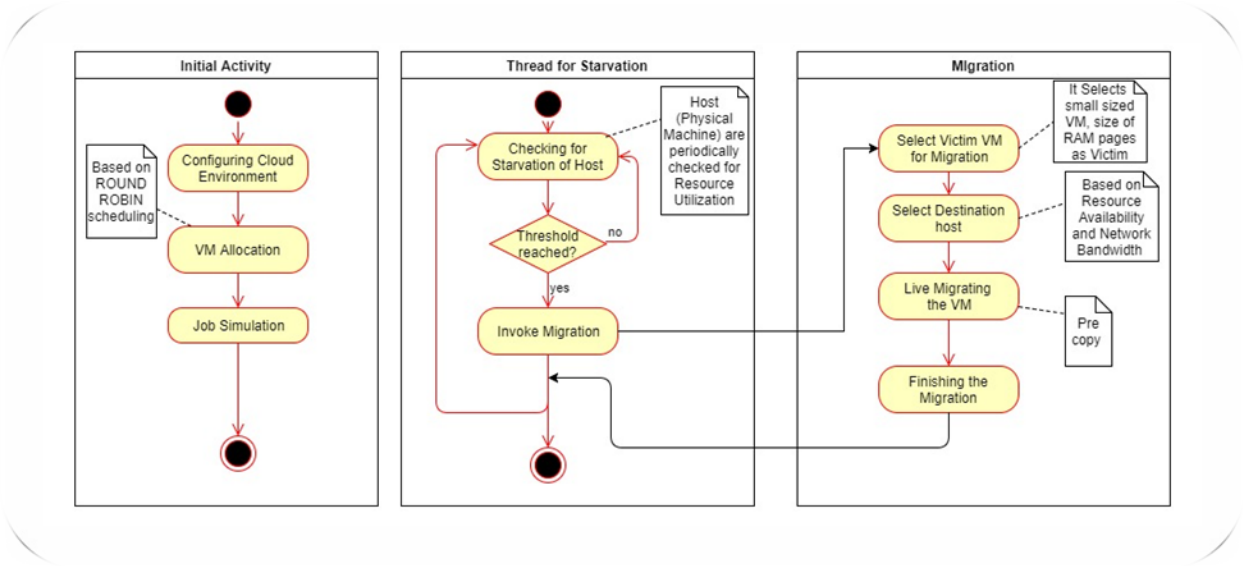

Initial Activity
Configuring Cloud Environment
Based on ROUND ROBIN scheduling
VM Allocation
Job Simulation
Thread for Starvation
Checking for Starvation of Host
Host (Physical Machine) are periodically checked for Resource Utilization
Threshold reached?
no
yes
Invoke Migration
MIgration
Select Victim VM for Migration
It Selects small sized VM, size of RAM pages as Victim
Select Destination host
Based on Resource Availability and Network Bandwidth
Live Migrating the VM
Pre copy
Finishing the Migration

Figure 2 - System process

The *System Process* activity diagram summarizes the activities listed below. Certain VMs are allocated in the host, this is the allocation activity and this done by the Round Robin way of scheduling based on the available resource, the resources reserved for that particular VM will be the allocated resource. Once the allocation of VMs is over jobs for each VM is done so that it will utilize some resources. The Monitor will do the starvation check activity by periodically checking the resource utilization of the host by the VMs, so that its threshold of different resources like storage, RAM, Mips, Pe is not reached. If it crosses the threshold, then there is a starvation of that particular host for resources. In order to avoid that we have to make space (resource available) in this host by migrating one more running VMs from the particular starving host to another available host. This is the Migration activity which involves selection of victim VM in the starving host and selection if the available and suitable destination host for the migration.

3.0. Algorithm

3.1 VM Allocation Algorithm

Certain VMs are allocated in the host, this is the allocation activity and this done by the Round Robin way of scheduling based on the available resource, the resources reserved for that particular VM will be the allocated resource. Once the allocation of VMs is over jobs for each VM is done so that it will utilize some resources.

```
      foreach host in hostList do
      {
//checking the host Storage,MIPS,RAM
      if((x1=H_list.get(i).get_host_fram())>=r&&(x2=H_list.get(i)
      .get_host_fstorage())>=s&&(x3=H_list.get(i).get_host_fmips(
      ))>=m)
      {
//If this host can occupy VM then this host is added to FV_list
            find_id[j]=i;
            String hf_id=H_list.get(i).get_host_id();
            FV_list.add(new FindVm(i,x2,x1,x3,hf_id));
            j++;
      }
      }
      if(FV_list.size()>1)
      {
            //Sorts the FV_list to insert vm in optimal best-fit
            host.
            Collections.sort(FV_list,FindVm.pComparator);
      }
//Inserting VM in top of FV_list.
```

3.2 Migration Algorithm

The Monitor will do the starvation check activity by periodically checking the resource utilization of the host by the VMs, so that its threshold of different resources like storage, RAM, Mips, Pe is not reached. If it crosses the threshold, then there is a starvation of that particular host for resources. In order to avoid that we have to make space (resource available) in this host by migrating one more running VMs from the particular starving host to another available host. This is the Migration activity which involves selection of victim VM in the starving host and selection if the available and suitable destination host for the migration.

```
//Round Robin based checking in host for starvation
            for(int i=0;i<H_list.size();i++)
      {
            String host_id=H_list.get(i).get_host_id();
      float h_ram=(float)((0.80)*(H_list.get(i).get_host_ram()));
            float
      h_st=(float)((0.90)*(H_list.get(i).get_host_storage()));
            float
      h_mips=(float)((0.75)*(H_list.get(i).get_host_mips()));
//80% of ram,90% of storage,75% of Processor**
            if(olst>=h_st)
            {
                  System.out.println(host_id+" is Overloading in
      Storage...");
            }
            if(olmips>=h_m)
            {
                  System.out.println(host_id+" is Overloading in
      Processing Element...");
            }
            if(olram>=h_ram|| olst>=h_st || olmips>=h_m)
            {
//live migration started in that host
                  liveMigrate(host_id);
            }
      }
```

3.2.1 Selection Of Victim VM:

```
//Adding all vm to LV_list from the victim host
      for(int i=0;i<V_list.size();i++)
      {
            if((q=V_list.get(i).get_host_id()).equals(h_id))
            {
                  LV_list.add(V_list.get(i));
            }
      }
//Sorting all vm’s and selects smallest vm to migrate
      Collections.sort(LV_list);
```

3.2.2 Selection Of Target Host:

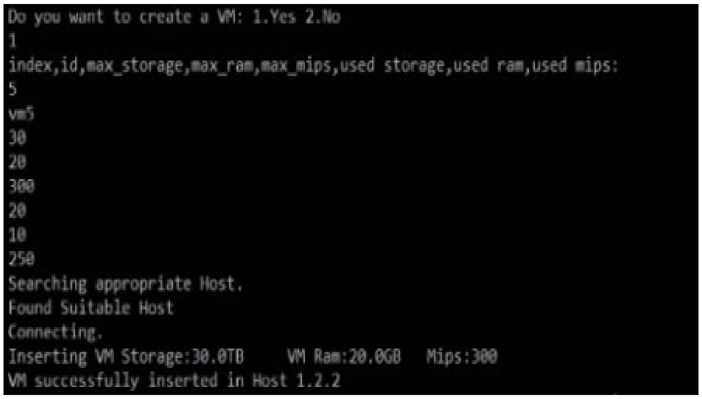
```
//Finding suitable set of Hosts
      for(int i=0;i<H_list.size();i++)
      {
            if(((h_r=H_list.get(i).get_host_ram())>ram)&&((h_s=H_l
      ist.get(i).get_host_storage())>storage)&&((h_m=H_list.get(i
      ).get_host_mips())>mips))
            {
                  Find_H_list.add(H_list.get(i));
            }
      }



//Sorting with respect to bandwidth
      Collections.sort(Find_H_list);
      Collections.reverse(Find_H_list);
      String hd=Find_H_list.get(0).get_host_id();
      int found=-1;
      for(int i=0;i<H_list.size();i++)
      {
            if((q=H_list.get(i).get_host_id()).equals(hd))
            {
                  found=i;
                  break;
      }
//Live Migration Started
      }
```

3.2.3 Live Migration Stages:

- **Stage0: Pre-Migration** We begin with an active VM on physical host A. To speed any future migration, a target host may be preselected where the resources required to receive migration will be guaranteed.
- **Stage1: Reservation** A request is issued to migrate an OS from host A to host B. We initially confirm that the necessary resources are available on B and reserve a VM container of that size. Failure to secure resources here means that the VM simply continues to run on A unaffected.
- **Stage2: Iterative Pre-Copy** During the first iteration, all pages are transferred from A to B. Subsequent iterations copy only those pages dirtied during the previous transfer phase.
- **Stage3: Stop-and-Copy** We suspend the running OS instance at A and redirect its network traffic to B. As described earlier, CPU state and any remaining inconsistent memory pages are then transferred. At the end of this stage there is a consistent suspended copy of the VM at both A and B. The copy at A is still considered to be primary and is resumed in case of failure.
- **Stage4: Commitment** Host B indicates to A that it has successfully received a consistent OS image. Host A acknowledges this message as commitment of the migration transaction: host A may now discard the original VM, and host B becomes the primary host.
- **Stage5: Activation** The migrated VM on B is now activated. Post-migration code runs to reattach device drivers to the new machine and advertise moved IP addresses.

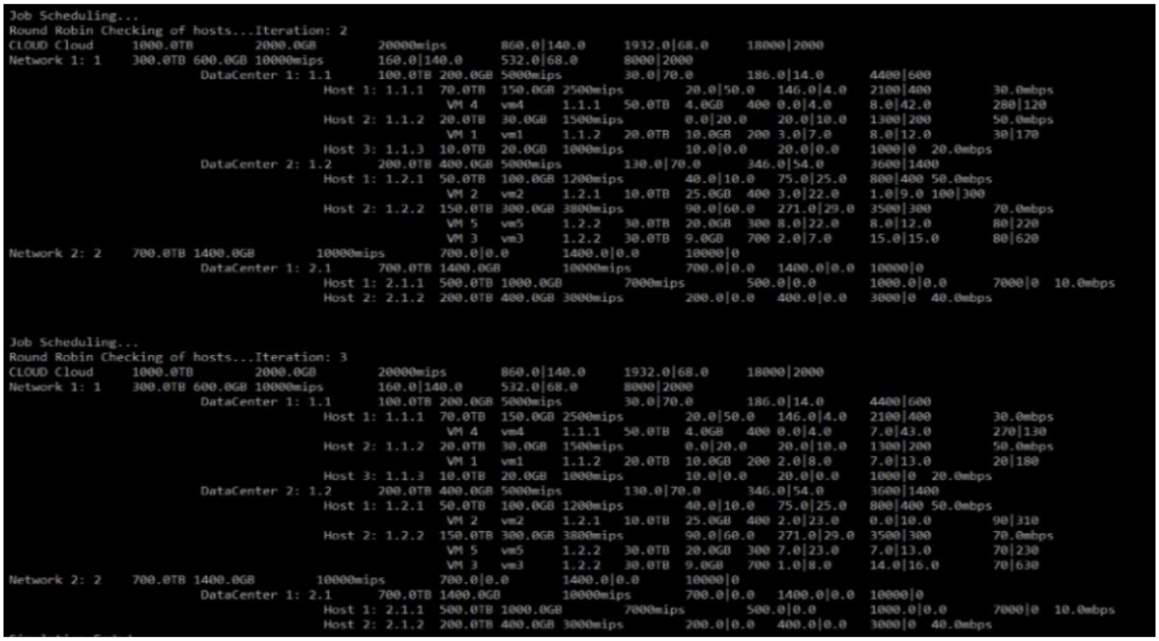
Screenshots:

Allocation Of VM



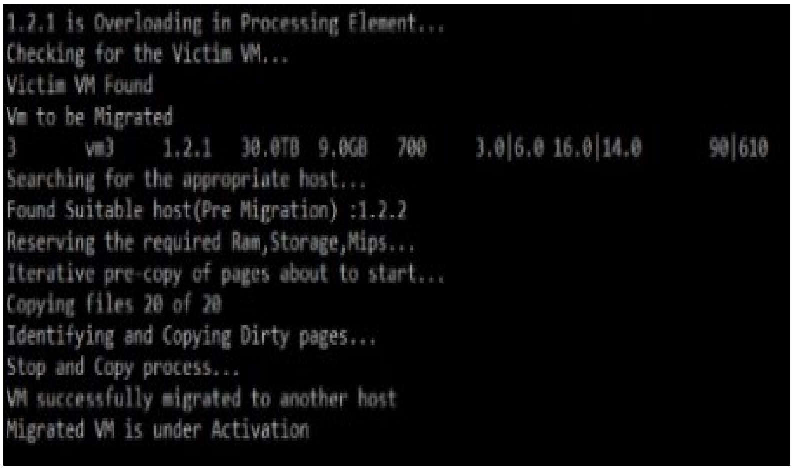
```
Do you want to create a VM: 1.Yes 2.No
1
index,id,max_storage,max_ram,max_mips,used storage,used ram,used mips:
5
vm5
30
20
300
20
10
250
Searching appropriate Host.
Found Suitable Host
Connecting.
Inserting VM Storage:30.0TB     VM Ram:20.0GB   Mips:300
VM successfully inserted in Host 1.2.2
```

Job Simulation



```
Job Scheduling...
Round Robin Checking of hosts...Iteration: 2
CLOUD Cloud         1000.0TB            2000.0GB            20000mips           860.0|140.0         1932.0|68.0         18000|2000
Network 1: 1        300.0TB 600.0GB 10000mips               160.0|140.0         532.0|68.0          8000|2000
                               DataCenter 1: 1.1            100.0TB 200.0GB 5000mips                30.0|70.0           186.0|14.0          4400|600
                                                   Host 1: 1.1.1  70.0TB   150.0GB 2500mips                  20.0|50.0   146.0|4.0    2100|400            30.0mbps
                                                                  VM 4   vm4      1.1.1   50.0TB  4.0GB   400 0.0|4.0    8.0|42.0            280|120
                                                   Host 2: 1.1.2  20.0TB   30.0GB  1500mips                  0.0|20.0    20.0|10.0    1300|200            50.0mbps
                                                                  VM 1   vm1      1.1.2   20.0TB  10.0GB  200 3.0|7.0    8.0|12.0            30|170
                                                   Host 3: 1.1.3  10.0TB   20.0GB  1000mips                  10.0|0.0    20.0|0.0     1000|0  20.0mbps
                               DataCenter 2: 1.2            200.0TB 400.0GB 5000mips                130.0|70.0          346.0|54.0          3600|1400
                                                   Host 1: 1.2.1  50.0TB   100.0GB 1200mips                  40.0|10.0   75.0|25.0    800|400 50.0mbps
                                                                  VM 2   vm2      1.2.1   10.0TB  25.0GB  400 3.0|22.0   1.0|9.0 100|300
                                                   Host 2: 1.2.2  150.0TB  300.0GB 3800mips                  90.0|60.0   271.0|29.0   3500|300            70.0mbps
                                                                  VM 5   vm5      1.2.2   30.0TB  20.0GB  300 8.0|22.0   8.0|12.0            80|220
                                                                  VM 3   vm3      1.2.2   30.0TB  9.0GB   700 2.0|7.0    15.0|15.0           80|620
Network 2: 2        700.0TB 1400.0GB          10000mips           700.0|0.0           1400.0|0.0          10000|0
                               DataCenter 1: 2.1            700.0TB 1400.0GB            10000mips           700.0|0.0   1400.0|0.0   10000|0
                                                   Host 1: 2.1.1  500.0TB 1000.0GB            7000mips            500.0|0.0           1000.0|0.0          7000|0  10.0mbps
                                                   Host 2: 2.1.2  200.0TB 400.0GB 3000mips                  200.0|0.0   400.0|0.0    3000|0  40.0mbps


Job Scheduling...
Round Robin Checking of hosts...Iteration: 3
CLOUD Cloud         1000.0TB            2000.0GB            20000mips           860.0|140.0         1932.0|68.0         18000|2000
Network 1: 1        300.0TB 600.0GB 10000mips               160.0|140.0         532.0|68.0          8000|2000
                               DataCenter 1: 1.1            100.0TB 200.0GB 5000mips                30.0|70.0           186.0|14.0          4400|600
                                                   Host 1: 1.1.1  70.0TB   150.0GB 2500mips                  20.0|50.0   146.0|4.0    2100|400            30.0mbps
                                                                  VM 4   vm4      1.1.1   50.0TB  4.0GB   400 0.0|4.0    7.0|43.0            270|130
                                                   Host 2: 1.1.2  20.0TB   30.0GB  1500mips                  0.0|20.0    20.0|10.0    1300|200            50.0mbps
                                                                  VM 1   vm1      1.1.2   20.0TB  10.0GB  200 2.0|8.0    7.0|13.0            20|180
                                                   Host 3: 1.1.3  10.0TB   20.0GB  1000mips                  10.0|0.0    20.0|0.0     1000|0  20.0mbps
                               DataCenter 2: 1.2            200.0TB 400.0GB 5000mips                130.0|70.0          346.0|54.0          3600|1400
                                                   Host 1: 1.2.1  50.0TB   100.0GB 1200mips                  40.0|10.0   75.0|25.0    800|400 50.0mbps
                                                                  VM 2   vm2      1.2.1   10.0TB  25.0GB  400 2.0|23.0   0.0|10.0            90|310
                                                   Host 2: 1.2.2  150.0TB  300.0GB 3800mips                  90.0|60.0   271.0|29.0   3500|300            70.0mbps
                                                                  VM 5   vm5      1.2.2   30.0TB  20.0GB  300 7.0|23.0   7.0|13.0            70|230
                                                                  VM 3   vm3      1.2.2   30.0TB  9.0GB   700 1.0|8.0    14.0|16.0           70|630
Network 2: 2        700.0TB 1400.0GB          10000mips           700.0|0.0           1400.0|0.0          10000|0
                               DataCenter 1: 2.1            700.0TB 1400.0GB            10000mips           700.0|0.0   1400.0|0.0   10000|0
                                                   Host 1: 2.1.1  500.0TB 1000.0GB            7000mips            500.0|0.0           1000.0|0.0          7000|0  10.0mbps
                                                   Host 2: 2.1.2  200.0TB 400.0GB 3000mips                  200.0|0.0   400.0|0.0    3000|0  40.0mbps
```

VM And Host Selection for Live Migration



```
1.2.1 is Overloading in Processing Element...
Checking for the Victim VM...
Victim VM Found
Vm to be Migrated
3       vm3     1.2.1   30.0TB  9.0GB   700     3.0|6.0 16.0|14.0       90|610
Searching for the appropriate host...
Found Suitable host(Pre Migration) :1.2.2
Reserving the required Ram,Storage,Mips...
Iterative pre-copy of pages about to start...
Copying files 20 of 20
Identifying and Copying Dirty pages...
Stop and Copy process...
VM successfully migrated to another host
Migrated VM is under Activation
```

4.0.References:

[1] Christopher Clark, Keir Fraser, Steven Hand, Jacob Gorm Hansen, Eric Jul, Christian Limpach, Ian Pratt, Andrew Warfield: Live Migration of Virtual Machines .

[2] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov,C ´ esar A. F. De Rose and Rajkumar Buyya: CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms(24 August 2010 in Wiley Online Library).

[3] Narander Kumar,Swati Saxena: Migration Performance Of Cloud Applications- A Quantitative Analysis(ICACTA-2015).

[4] Heba Kurdi,Ebtesam Aloboud,Sarah Alhassan,Ebtehal T.Alotaibi: An Algorithm For Handling Starvation and Resource Rejection In Public Clouds(ICFNC-2014).