Advanced Load Balancing in Baadal

Group 1
Mayank Singh Chauhan 2016CS50394
Atishya Jain 2016CS50393
Mankaran Singh 2016CS50391
Avaljot Singh 2016CS50389

November 15, 2019

1 Introduction

Baadal is a cloud orchestration software built for IITD private cloud for high performance computing. It is based on open technologies like Linux, KVM, libvirt, OpenVSwitch, Apache, MySQL and web2py.
The main features of baadal include dynamic resource scheduling and power management and dynamic resource utilisation monitoring. It also provides facilities for suspend, resume, shutdown, power off, power on and specifying resource requirement of virtual machines.
The main purpose for this assignment is to design, develop and integrate a load balancing module in Baadal. This includes studying the existing code of python, collecting load statistics and display it on dashboard, creating a migration plan based on load statistics and then actually migrate a VM according to migration plan.
Load Statistics are based on 3 things:

1. CPU and RAM Utilisation for Hosts and VMs.
2. Disk Load attached to VMs for Read and Write.
3. Network Traffic of VMs for Read and Write.

2 Installation of Baadal

We were able to install Baadal on VMware in our personnel Laptop. The host machine had 16GB RAM, 1TB SSD, 8 core processor and Windows 10 OS. The VM we created using VMware had 8GB RAM, 10 GB Hard disk, 4 core processor and Ubuntu 16.04 OS.

2.1 Steps to install baadal:

- Set environment variables for proxy server.

```
export http_proxy="proxy62.iitd.ac.in:3128"
export https_proxy="proxy62.iitd.ac.in:3128"
```

- Update Repositories Source list.

```
sudo apt-get update
sudo apt-get upgrade
```

- Install git and clone baadal repository from github.

```
sudo apt-get install git
git clone https://github.com/iitd-plos/baadal2.0
```

- Install Baadal.

```
cd /baadal2.0/baadaltesting/devbox
sudo -E make devbox
```

It took 12 min to complete the installation. It is possible that the installation throws up errors due to inability of dependencies to get resolved. It is important to reboot and repeat this step every time such errors occur.

- Possible errors:-

```
If libvirt is not able to acquire libvirtd.pid, manually delete that file and make again
If libvirt-socket is not accessible, manually give that file a permission of 777.
```

- To run the scheduler, use the following commands:

```
cd /home/www-data/web2py
su www-data -c "python web2py.py -K baadal:vm_task,baadal:vm_sanity,baadal:host_task,
baadal:vm_rrd,baadal:snapshot_task &"
```

- Now we can login as admin with Username **admin** and password **baadal**. This will allow us to request for new VM's and approve the requests.

- When you create a new VM, you can get access to it by the command:

```
sudo virsh console <vm name here>
```

The default username is cirros and password is cubswin:)

3 DashBoard Design

For each host and VM, we will be monitoring the following:

- Memory Usage Percentage
- CPU Usage Percentage
- Disk Reads and Writes
- Network Reads and Writes

3.1 Analysis of load on a VM

After modifying the code, we can see load statistics for the VM under the **Dashboard Module 7x : Per VM** tab.

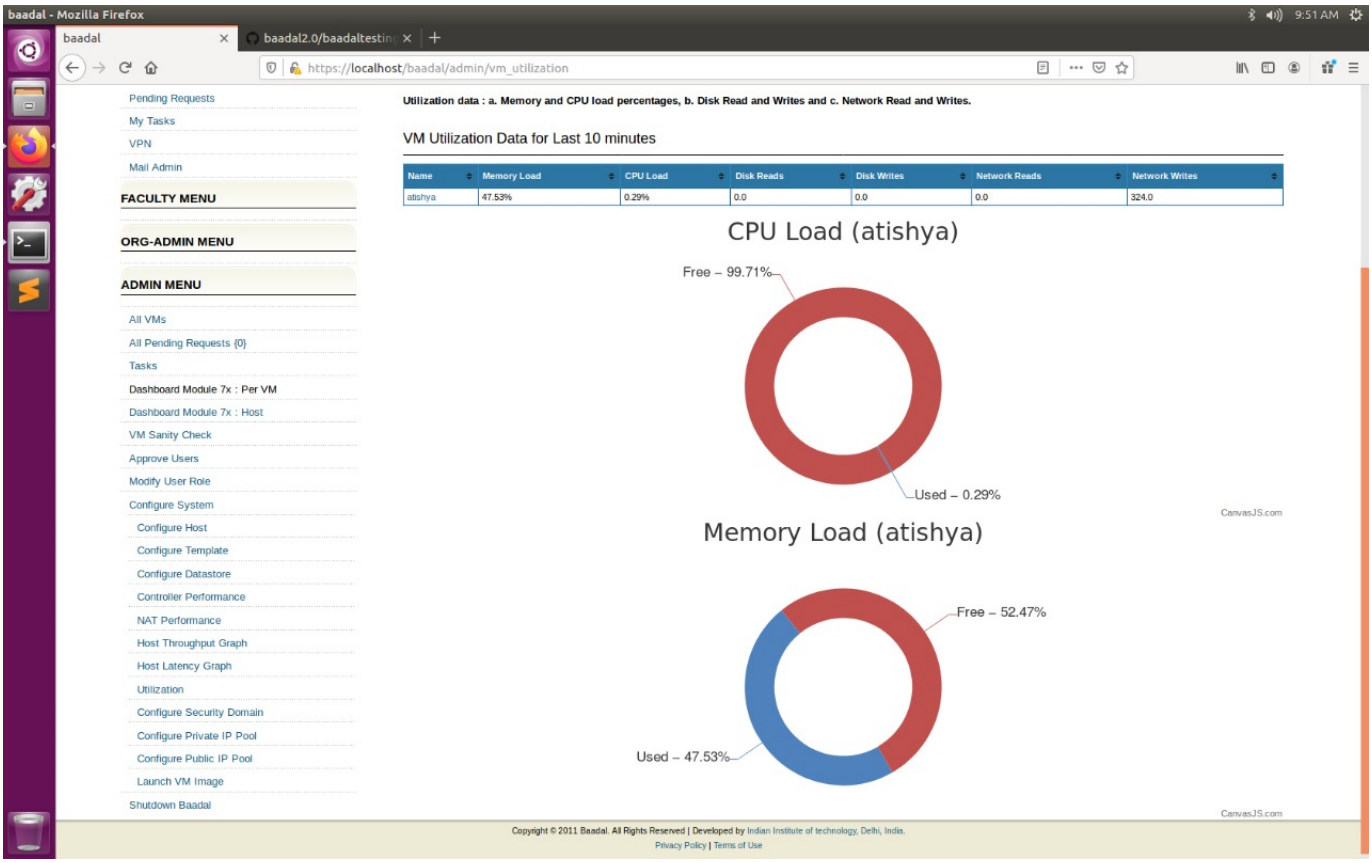

Figure 1: CPU and Memory Usage Percentage

The above figure shows the percentage CPU and Disk usage by the VM in form of circular pie charts. In this way, we can display the percentage CPU and Disk usage by all VMs. We can see the exact RAM also. Also, VM's with more than 80% load are marked in Red.

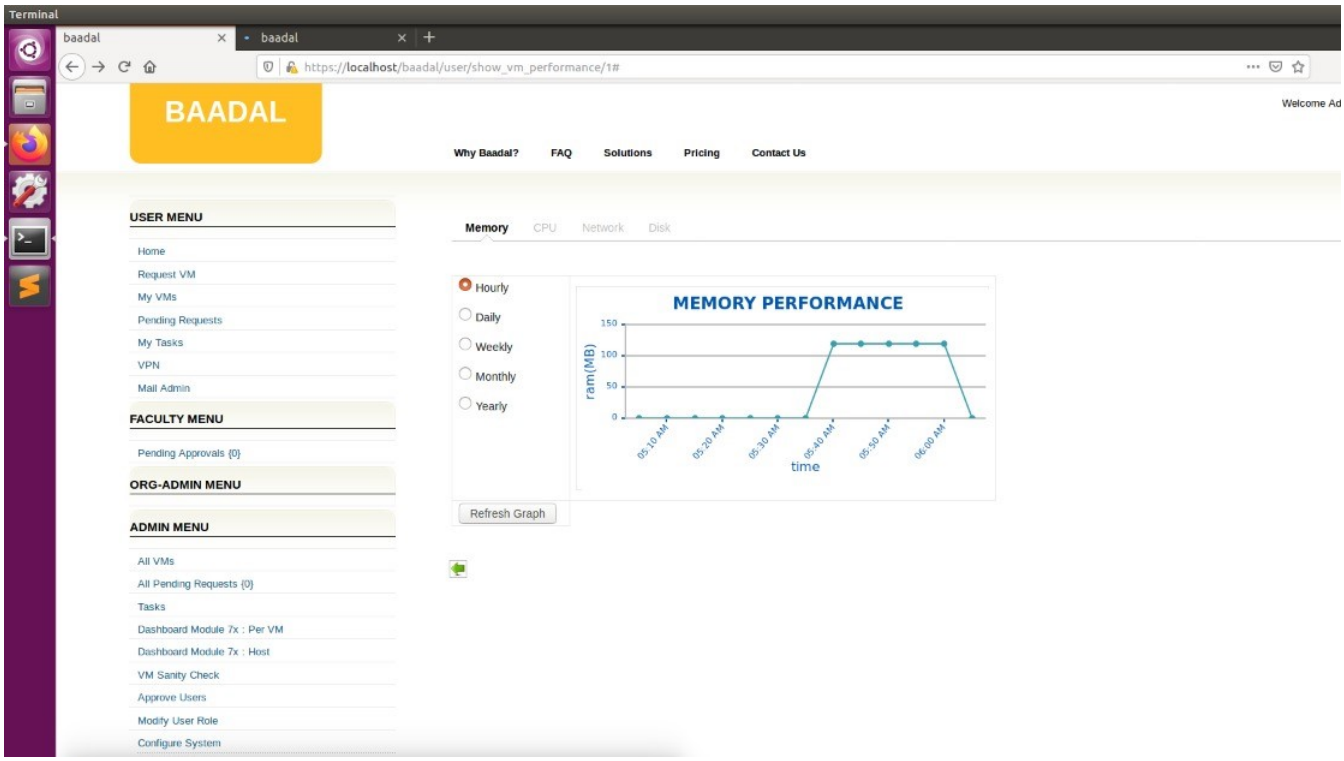

Figure 2: Memory Usage

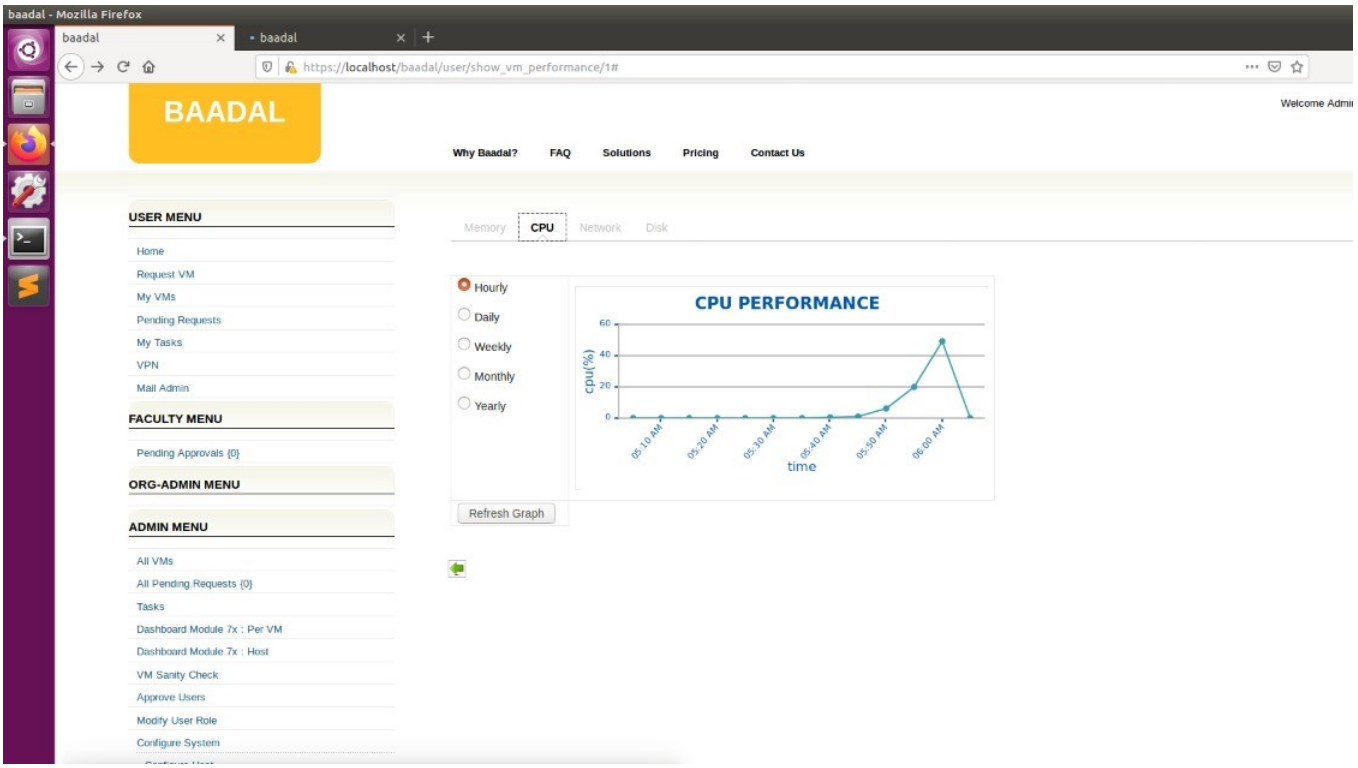

Figure 3: CPU Usage

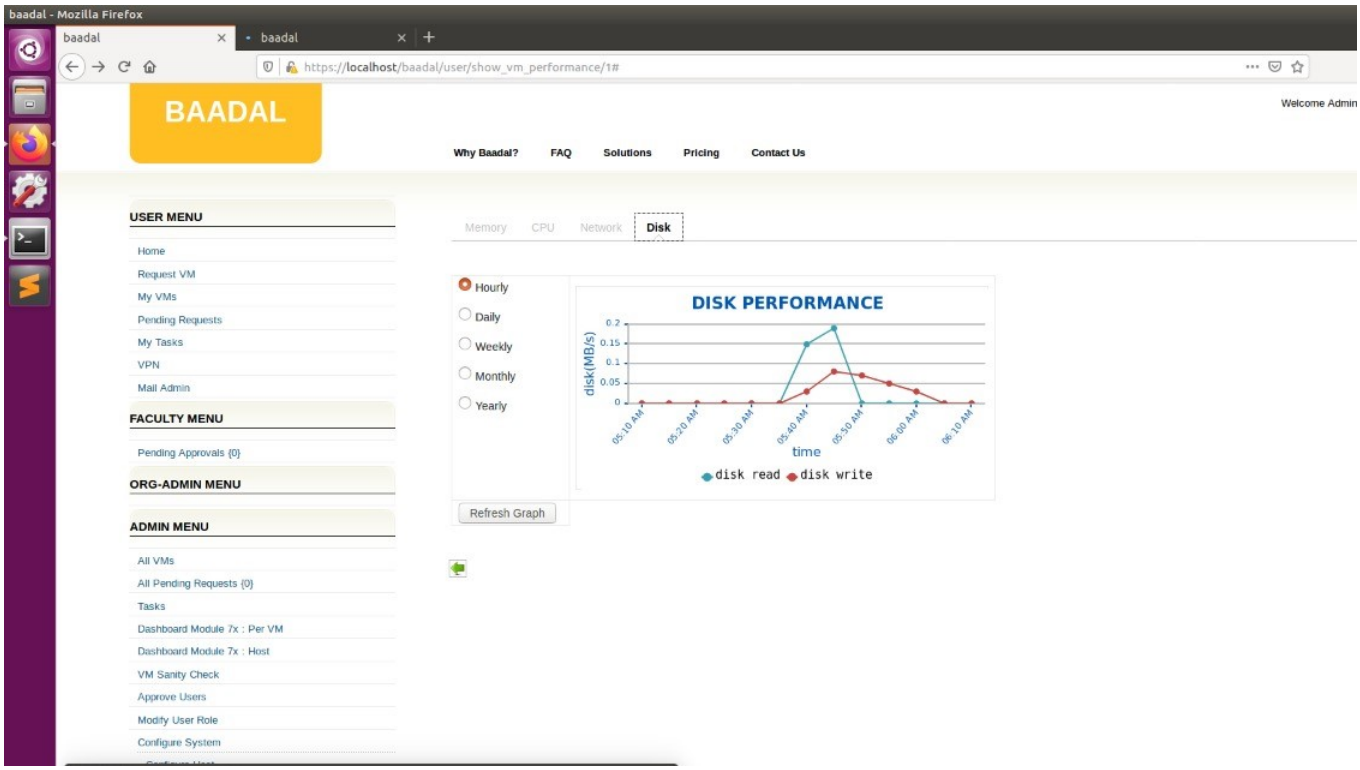

Figure 4: Disk Usage

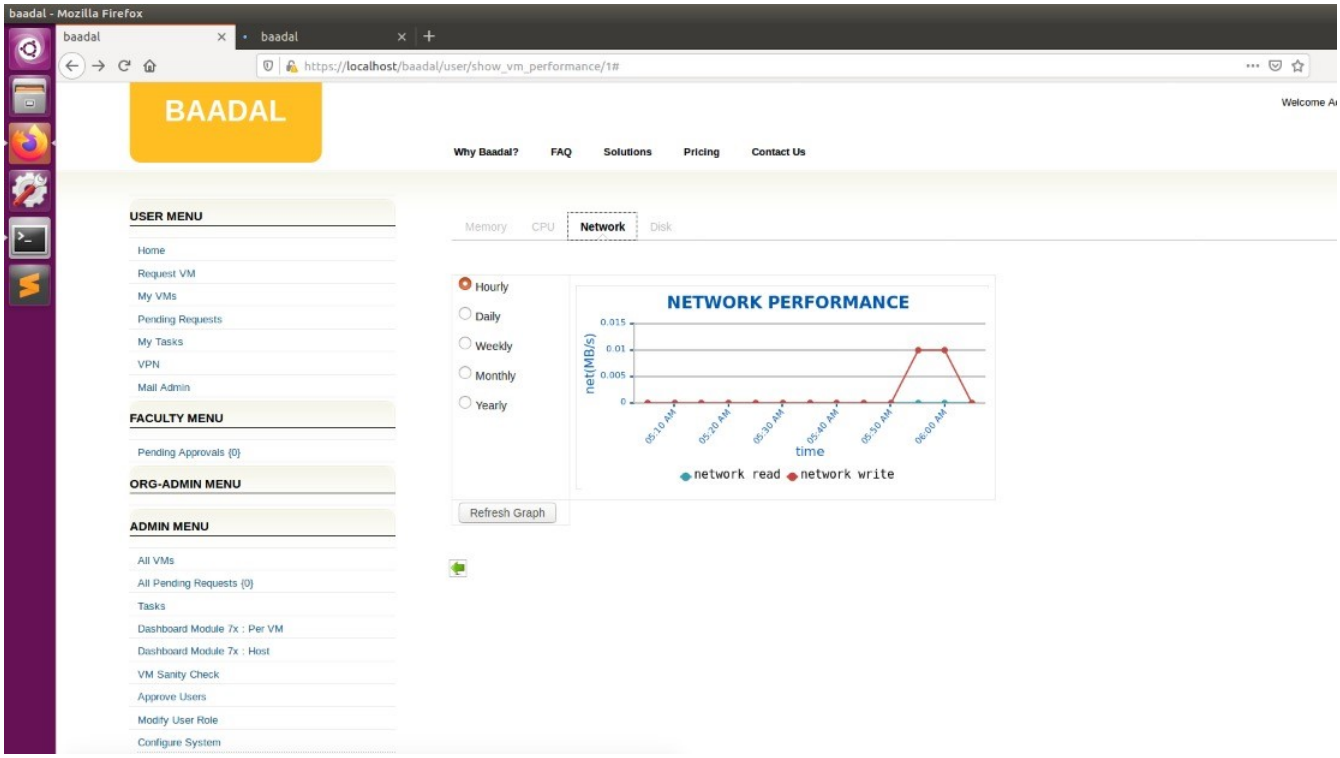

Figure 5: Network Usage

3.2 Analysis of load on a Host

We can see load statistics for the Host under the **Dashboard Module 7x : Host** tab.

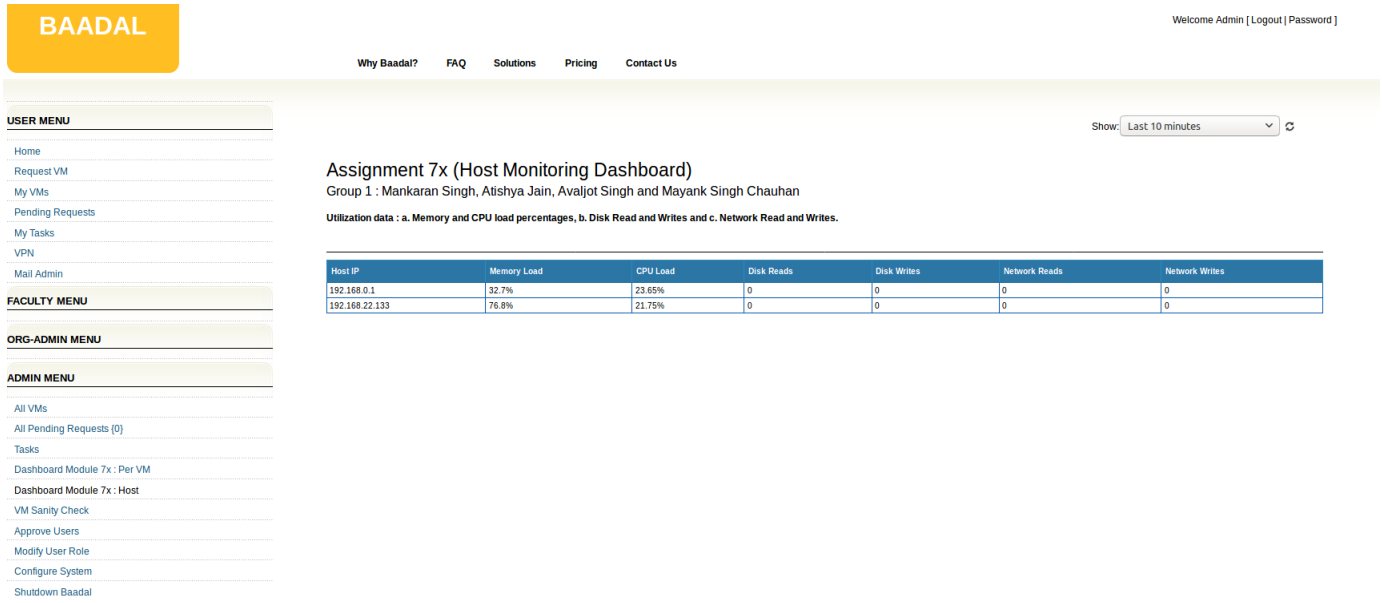

Figure 6: Host Load Stats

It basically displays Memory used, CPU load, Disk reads & writes and Network reads & writes for all the Hosts in a Tabular form. From here we can analyse the load on each Host separately.

4 Migration

Migration is required for Load Balancing. When a host faces a large amount of load, then a VM hosted on it can be migrated to another host with less load to balance load on both hosts. For this we need to decide which VM on a host should be moved to another host based on load statistics. This is called formulating a Migration Plan. After that, we will migrate the VM following this plan.

4.1 Formulating a Migration Plan

For formulating the migration plan of the VMs across different hosts, let us assume that in total m VMs are are hosted by n hosts ($m > n$). Also, we consider 4 different types of load parameters, namely:

1. CPU load; l_{CPU}

2. Memory load; l_{Mem}
3. Storage load; l_{St}
4. Network load; l_{Net}

These loads will be shared by the hosts such that the following equations hold:

$$\sum_{i=1}^{n} l_{CPU}(i) = \mu_{CPU}.m$$

$$\sum_{i=1}^{n} l_{Mem}(i) = \mu_{Mem}.m$$

$$\sum_{i=1}^{n} l_{St}(i) = \mu_{St}.m$$

$$\sum_{i=1}^{n} l_{Net}(i) = \mu_{Net}.m$$

where μ_{CPU}, μ_{Mem}, μ_{St}, and μ_{Net} are the constants of proportionality. Since most of the load faced by the host is processor load followed by Memory, Storage and Network load, we can intuitively say, $\mu_{CPU} > \mu_{Mem} > \mu_{St} > \mu_{Net}$.
The idea is that whenever the load on a particular host goes above a certain threshold, we need to migrate the VMs hosted by the particular host to another host. So, we define the total load faced by a host which is the weighted combination of the 4 different loads, i.e.,

$$l(i) = \lambda_{CPU}.l_{CPU}(i) + \lambda_{Mem}.l_{Mem}(i) + \lambda_{St}.l_{St}(i) + \lambda_{Net}.l_{Net}(i);$$

$$\lambda_{CPU} + \lambda_{Mem} + \lambda_{St} + \lambda_{Net} = 1$$

When the load of a VM increases above θ_{max}, we need to migrate its VMs to another host which has a lesser load. The parameter θ_{max} can be configured.

4.2 Migrating the VM

Before we can even think of migrating the VM from one host to another, we will have to somehow connect the two hosts which initially are two inpendent systems. VMs in devbox are created in 192.168.0.0/16 private subnet. So, VMs running on one devbox cannot connect to VM running in another devbox. Follow the below steps to do that:

1. Controller of only one devbox should be used. Other devbox machines should be used only as host machines to avoid any data conflict.
2. Each VM will use the DHCP Server running on its own devbox server. So, if private IPs(other than host IPs) are added to the controller; DHCP configuration file (/etc/dhcp/dhcpd.conf) on controller needs to be copied to other host machines; and DHCP server should be restarted using following command.

```
service isc-dhcp-server restart
```

3. There should be common datastores on all machines to facilitate VM migration. To mount Controller datastore on other machines, edit /etc/exports to give permission to host IPs. Sample edited entry will be as follows:

```
/baadal/data 10.17.0.0/16(rw,sync,no_root_squash,no_all_squash,subtree_check)
```

4. Edit /etc/fstab on host machines to mount file from controller datastore. Sample edited entry will be as follows:

```
10.17.6.41:/baadal/data /mnt/datastore nfs rw,auto
```

5. Use mount -a to mount the datastore on host machines

6. Passwordless ssh connection should be configured between controller and host machines; and also between host machines for migration. Execute following command on Controller. HOST_IP should be replaced with actual IP:

```
ssh-copy-id root@HOST_IP
su - www-data
ssh-copy-id root@HOST_IP
```

Execute following command on Host machines for each host including controller.

```
ssh-copy-id root@CONTROLLER_IP
```

7. Login to baadal web interface. Goto ADMIN MENU > Configure System > Configure Private IP Pool. Add entry of private ip and mac address of host machines in vlan0.

8. Goto ADMIN MENU > Configure System > Configure Host. Enter the host machine IP, and Click Get Details . Host Configuration details should get populated. Host can then be added to the system.

After doing this you'll see the second host added to your controller. To migrate a VM to another host, click on the settings icon next to the VM in View all VMs. Then you'll see an option Migrate this VM. On clicking that you'll see a menu as in the following image.

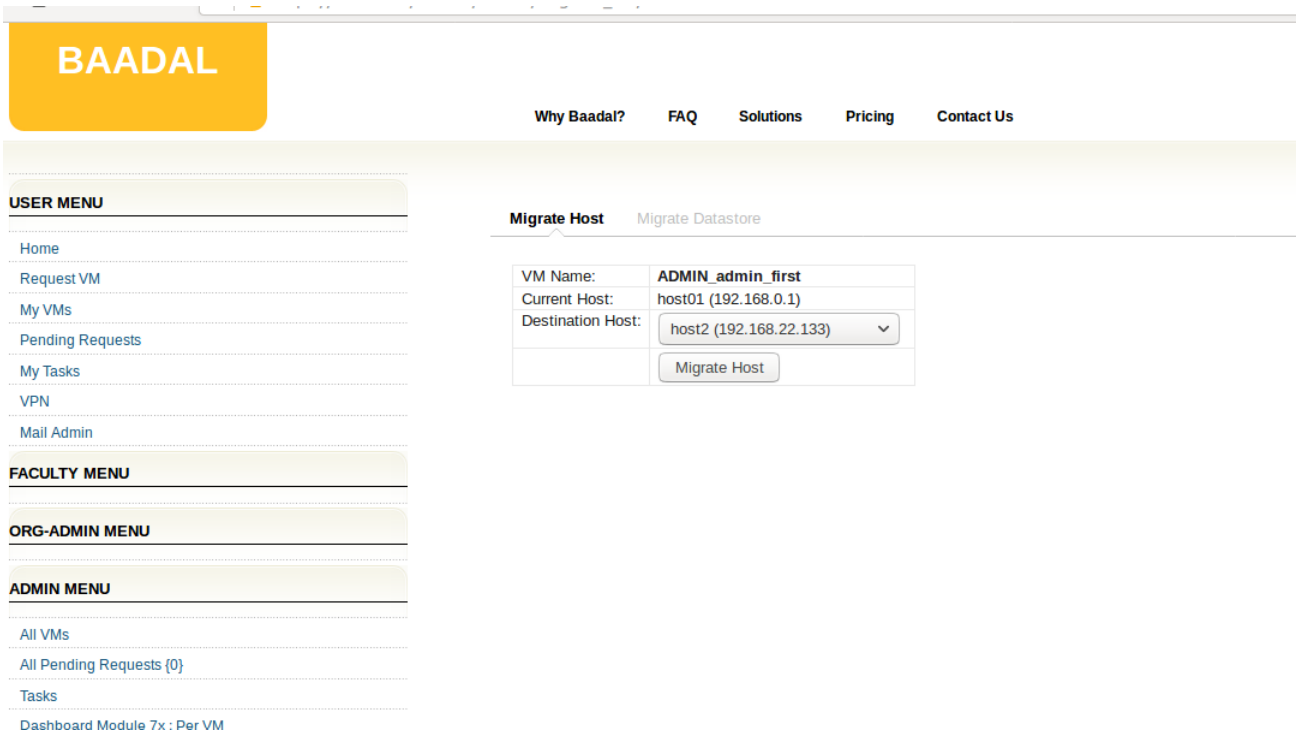

Figure 7: Migrating a VM

5 Implementation Details and Code Changes

We have integrated the dashboard functionality with the Baadal UI using web2py framework. Web2py uses the Model View Controller design pattern to implement the website. It ensures easy reusability and modifications on the application.

5.1 Models

The models manage all the data and state, independent of the UI. All the information about the VMs' cpu, memory, etc, as well as about the hosts' cpu, etc is stored in a Round Robin database periodically. In the models code, we modified the following files:

- **model_admin.py**
  The time period for which the user wants the statistics is taken as input, and sent to the model, so as to fetch data accordingly from the RRD database. The data for the last few minutes till several years was made available.
  get_host_util_data : This function was modified so that now it returns an object containing memory, cpu, disk reads, disk writes, network reads and network writes for each host. (This info was already being returned by the fetch_rrd_data function)
  get_vm_util_data : This function was modified so as to also return the RAM and Identity of each VM.

- **menu.py**
  Changes in menu.py file are done to add the buttons to the admin menu, since the UI

fetches the list of buttons from this file. 'Dashboard Module 7x: Host' button added to the Admin menu. It calls the 'hosts_vms' API.'Dashboard Module 7x: Per VM' button added to the same menu. It calls the 'vm_utilization' API. Both these API's are there in controllers/admin.py.

5.2 View

The view is actually manages the UI part. It consists of HTML pages, which present information to the user. The HTML files were updated to add the extra functionalities ad render the graphs or pie charts as per the user requirements. CanvasJS was used to build and render all the bar pie charts.

- **admin/hosts_vms.html**
  This is the file that is displayed on clicking on the Dashboard Module 7x: Host button. This was modified to display our group information. A custom table was added to show all the statistics of CPU, Memory of different hosts on this page. Also, pie charts - 2 per host, that display the hosts' relative percentage of free memory and CPU are also displayed to build a more intuitive interface for the users.

- **admin/vm_utilization.html**
  This is the page displayed on clicking on the Dashboard Module 7x: Per VM button. The existing table was modified to also display the identity of the VMs, which allows for easy identification of the VMs. Every vm Id is clickable, which takes the user to a separate page.

- **users/show_vm_performance.html**
  This is the page which is displayed when the user clicks on a VM Id. Here, we provide the user with 4 different tabs, one for each of Memory, CPU, Disk and Network stats. Under each tab, the user has the option of viewing an area graph with information about the usage/numbers for different time durations. The options of hourly, daily, weekly, monthly, yearly, as well as the option of selecting a time period on a particular date is provided.

5.3 Controller

The controller forms the bridge between the model and the view. It updates the other when one changes. The application logic lies mostly in the controllers. In the controllers code, we only modified the file admin.py.
In this file, the API host_vms was modified to return all the information like, CPU usage, Memory usage, disk and network read-writes about the host to the view. This API is actually called by UI when user clicks on **Dashboard Module 7x : Host**. Similarly, the API vm_utilization was modified to display VMs load statistics.

5.4 Modules

The file modules/vm_utilization.py contains the functions that collect the CPU, Memory, Disk Network statistics of all the VMs. The function fetch_rrd_data in this file is used by

the model files to get the stats. The function rrd_database_lookup() is used for querying the stats for different time durations.