Process Migration Using Docker Container in SDN Environment

Ashis Talukder, Sarder Fakhrul Abedin, Anupam Kumar Bairagi, Md Golam Rabiul Alam,

SeonHyeok Kim, Sungwon Lee and Choong Seon Hong

Kyung Hee University

Abstract

Energy consumption and load balancing are considered to be crucial research issue in Software Defined Networking (SDN). In this research we have tried to develop a process migration technique using docker container in SDN environment that will provide features like dynamic resource sharing, load balancing, flexibility, security, and fault tolerance. This migration process takes the advantage of docker such that lightweight, security, open and easy to develop. We have proposed an algorithm that will select processes if migration is necessary and then select the destination physical machines and assign processes to them by stable matching and finally migrate the process using docker container. Our algorithm reduces energy consumption and achieves load balancing.

1. Introduction

Process migration and Virtual Machine (VM) migration have been considered to be one of the important technologies in improving data center efficiency, such as reducing energy cost and balancing load [3]. With the development of Software Defined Networking (SDN), the process migration and VM migration are getting importance in SDN as well. In this research we have tried to develop an algorithm to migrate process which are running on one machine in SDN environment to another machine using docker container.

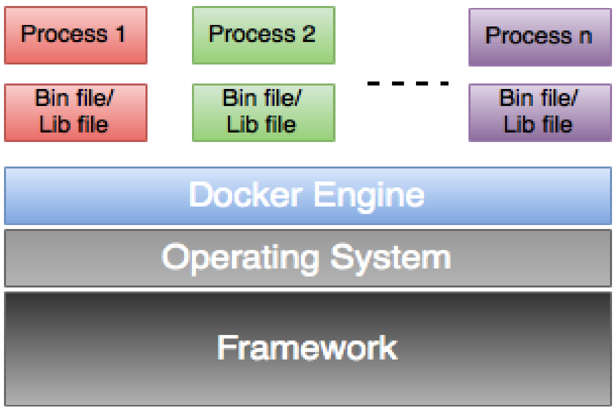

Figure 1: Running environment of Docker.

Docker containers can be run on any computer, udner any framework or infrastructure and in any cloud or SDN environment [6]. When the docker container is used in process migration the docker-image includes the complete filesystem that contains everything to run the process on the different machine like: code, runtime, system tools, and system libraries (see Figure 1). This ensures that the process will always run the same as before migration, regardless of the environment [6]. As compared to VM migration, the docker image does not need to include the guest operating system. That is the reason why docker is lightweight. Another salient features of docker include: it is open, it is secure.

The main contribution of this paper is to develop an Element Management System (EMS) module that will work with SDN controller. The EMS module that will migrate the process using docker container in SDN system, covers functions like (a) managing resources and application or processes, (b) dynamic balance of resource usage and load balancing, (c) flexible and efficient resource sharing, (d) fault tolerance, (e) portability, (f) resource consolidation etc. In this paper, we have proposed an algorithm for process migration where migration is a continuous process in the system and the algorithm works in three steps: (1) Select (2) Match (3) Migrate. The rest of the paper is organized like: in section two a literature review has been provided, the system model is given in the section three, evaluation and conclusion have been described in the next consecutive sections.

2. Related works

Previously research on process and VM migration was done in cloud environment but recently SDN is also considered in this area.

Wang et al. [1] tried to reduce the total migration tim e by determining the migration orders and transmiss ion rates of VMs. Zhang et al. [2] have developed a scheme to resume the network connection of the VM during VM migration and proposed a novel routing algorithm that achieves fast flow resumption on SDN.

In [4] the authors have defined the problem of minimizing the energy consumption ensuing QoS requirements in VM allocation policies for resource management in cloud data centers. Maziku et al. [5] have proposed VM management techniques for utilizing resources efficiently which reduces energy consumption and number of VM migration in virtualized data centers in cloud environment.

It is seen that the most of the techniques involve reducing energy consumption and maintaining load balance. Our algorithm also consider energy consumption in selection step and final migration achieves load balancing.

3. System Model

A) System Architecture

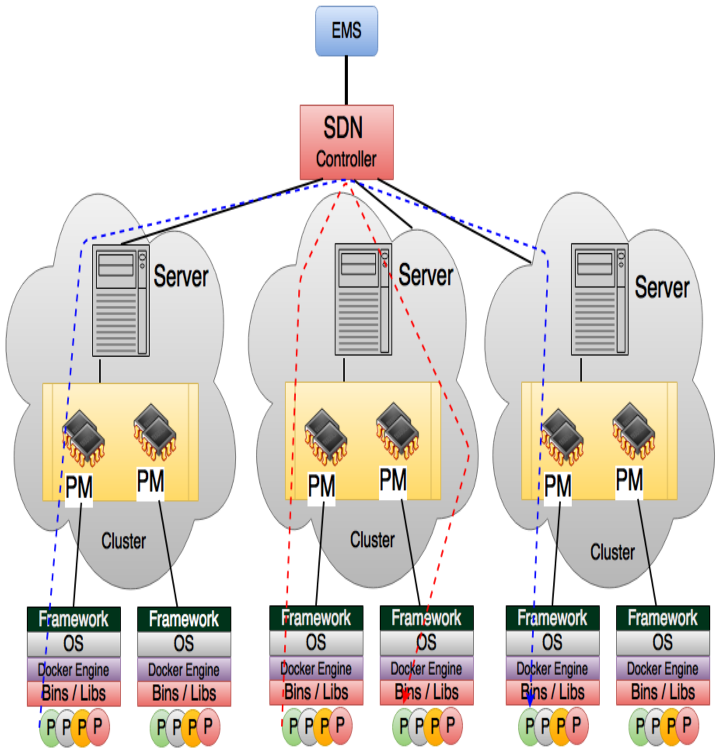

Figure 2: Migration model in SDN System

Consider a scenario of an Enterprise Network based on Software Defined Network (SDN) in Figure 2. As depicted in the figure, the SDN controller manages n servers $S = \{S_1, S_2, \dots, S_n\}$. Each server is equipped with a set of physical machines which are denoted as $PM = \{PM_1, PM_2, \dots, PM_k, PM_{k+1}, PM_{k+2}, \dots, PM_n\}$. User processes are actually running on these physical machines. We have proposed an Element Management System (EMS) that will run on SDN controller, collect the information about the servers, PM, and processes P, and finally instruct SDN controller about migration if necessary.

B) Migration process

Different studies show that the power consumption is linearly related to CPU utilization and an idle server uses about 70% energy that a fully utilized server uses [3]. Our migration process is initiated with collecting the energy information of the PMs by EMS (see Figure 3). We introduce a threshold to select the overloaded PMs. and thus, select processes with larger process ID (which have started running most recently).

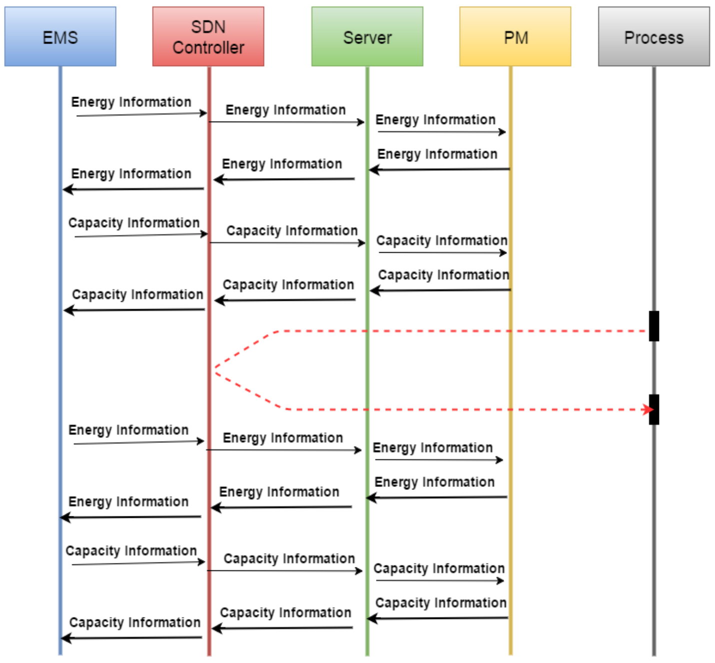

Figure 3: Sequence diagram

The EMS takes memory capacity information and then along with previous CPU utilization information and residual memory capacity information it selects a set of PMs to migrate the processes. Then stable matching is done to get the suitable migration. Finally processes Ps are migrated to the desired PMs using docker container. This process continues after a certain amount of time (see Algorithm 1).

```
Algorithm 1 Process Migration
1:  Get power information $P(u)$ and determine which $PMs$
    needed process migration with a threshold $P_\theta$
2:  if No migration is necessary then
3:      Go to Step 1 after a predefined time quantum
4:  else
5:      List the processes $p_i$ with higher $ID$ from $PMs$
6:  end if
7:  if No proper PM for migrated process then
8:      Go to Step 1 after a predefined time quantum
9:  else
10:     List the processes with higher $ID$ from $PMs$
11:     Make preferences list for $P_i$ based on the basis of
        energy and memory requirement
12:     Make preferences list for $PM - i$ based on residual
        energy and memory capacity
13:     Do the Process $p_i$ and $PM_i$ stable matching
14: end if
15: for each Process-PM couple $(P_i, PM_i)$ do
16:     Migrate the Process $p_i$ to the destination PM $PM_i$ using
        Docker Container.
17: end for
18: Go to Step 1 after a predefined time quantum
```

4. Evaluation

We have performed a simulation in Java on a machine (Core i3 2GHz, 3.5GHz, 8GB RAM, Windows 7) with synthesized data. We have created a list of processes P_i and PM_i and the requirements of CPU and memory space randomly. Then some of these have been selected for migration and some PMs were selected depending on their residual CPU and memory capacity. Then a preference list has been made (see Table 1 & 2). Then stable matching has been calculated for P – PM migration and listed in the Table 3. This process manages energy consumption and provides load balancing.

PM_i	P_i preferences
PM1	P1, P2, P3, P4, P5
PM2	P2, P5, P1, P3, P4
PM3	P4, P3, P2, P1, P5
PM4	P1, P2, P3, P4, P5
PM5	P5, P2, P3, P4, P1

Table 1: Preference list of PM

P_i	PM_i preferences
P1	PM5, PM3, PM4, PM1, PM2
P2	PM1, PM2, PM3, PM5, PM4
P3	PM4, PM5, PM3, PM2, PM1
P4	PM5, PM2, PM1, PM4, PM3
P5	PM2, PM1, PM4, PM3, PM5

Table 2: preference list of P

PM	P
PM4	P1
PM1	P2
PM5	P3
PM3	P4
PM4	P5

Table 3: Assignment of Ps to PMs for migration

5. Conclusion

In this paper, we have tried to develop an EMS module that will work together with SDN based networking system. The main drawback of this approach is that docker container does not support live migration. Even though the EMS will migrate process from one physical machine to another physical machine for providing many dynamic features like resource sharing, load balancing, flexibility, fault tolerance etc. in Software Defined Networking (SDN).

Acknowledgement: This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (B0190-16-2013, Development of Access Technology Agnostic Next-Generation Networking Technology for Wired-Wireless Converged Networks) *Dr. CS Hong is the corresponding author.

6. References

[1]. Wang, H., Li, Y., Zhang, Y. and Jin, D., 2015, April. Virtual machine migration planning in software-defined networks. In *Computer Communications (INFOCOM), 2015 IEEE Conference on* (pp. 487-495). IEEE.

[2]. Zhang, S.Q., Yasrebi, P., Tizghadam, A., Bannazadeh, H. and Leon-Garcia, A., 2015, November. Fast Network Flow Resumption for Live Virtual Machine Migration on SDN. In *2015 IEEE 23rd International Conference on Network Protocols (ICNP)* (pp. 446-452). IEEE.

[3]. Kella, A. and Belalem, G., 2014. VM Live Migration Algorithm Based on Stable Matching Model to Improve Energy Consumption and Quality of Service. In *CLOSER* (pp. 118-128).

[4]. Beloglazov, A. and Buyya, R., 2010, May. Energy efficient resource management in virtualized cloud data centers. In *Proceedings of the 2010 10th IEEE/ACM international conference on cluster, cloud and grid computing* (pp. 826-831). IEEE Computer Society.

[5]. Maziku, H. and Shetty, S., 2014, March. Network Aware VM Migration in Cloud Data Centers. In *Research and Educational Experiment Workshop (GREE), 2014 Third GENI* (pp. 25-28). IEEE.

[6]. https://docs.docker.com/