Linux Containers: An Emerging Cloud Technology

Asad Javed
Aalto University, School of Science
asad.javed@aalto.fi

Abstract—Linux containers, commonly known as LXC, has become a popular approach for creating virtual environment inside a large computer system. It is a lightweight virtualization infrastructure, which creates multiple virtual Linux systems simultaneously on a single host machine. Linux kernel features are used to provide isolated environment for applications within multiple systems. The main goal of LXC is to create environment as close as possible to the standard Linux system, thus providing scalable and performance efficient systems, thereby considering as an emerging cloud technology.

This paper presents a thorough survey about LXC along with two well-known and important kernel features including namespaces and cgroups. In this paper, the comparison between container-based and hypervisor-based virtualization is also been made in order to completely understand this phenomenon. At the end, Docker platform is explained which is an open source project and a running application of Linux containers.

Keywords—linux, namespaces, kernel, cloud, cgroups, docker

I. INTRODUCTION

In recent years, virtualization has become very popular in large distributed systems, since the need of making the whole system efficient become evident. With the advent of distributed systems and cloud computing, there happens to be a virtualized environment in almost every data centers. Due to this emergence, there are many software solutions such as VMware [1] which provides cloud and software services, Xen [2] that provides services based on a microkernel design, Kernel Virtual Machine (KVM) [3] that also provides full virtualization, and Hyper-V [4] which is specifically for windows server. These solutions are known as hypervisor-based virtualization and also considered efficient as they have many features. Some of the features includes portability, easier backups, isolation, security, and migration to cloud [5]. Due to large number of benefits, virtualization can also be considered as a foundation of cloud technology.

Although, virtualization provides efficiency and scalability but there is also a resource overhead when it comes to traditional virtualization. At that point, Linux container-based virtualization (LXC) come into practice. This approach has a different context as compared to hypervisor. It is the operating system level virtualization, unlike hypervisor which is a hardware-level virtualization, and creates multiple system on a single host. Conceptually, it can be considered as a lightweight virtualization because applications run in a separate container that shares the resources of the host operating system [5] [6]. The whole execution environment such as libraries, file system, scripts, application, and operating system are put in a container and then executed over a single host. Through that way, the system becomes more efficient in terms of processing, and applications run in an isolated form irrespective of the other applications [7] [8].

The rest of the paper is organized as follows. Section II describes the difference between traditional hypervisor-based virtualization and LXC approach. Section III presents kernel features which are used by containers in order to provide isolation and scalability. An application of LXC named Docker along with its usage will be illustrated in Section IV. Finally, the paper ends with a conclusion in Section V.

II. COMPARISON BETWEEN HYPERVISOR AND CONTAINERS

Despite the fact that virtualization offers huge amount of benefits, there is still a difference between hypervisor-based and container-based virtualization, where the latter one provides greater efficiency [8] [9]. The comparison is shown in Fig. 1.

![Fig. 1. Hypervisor vs Linux containers](image-url)
According to Fig. 1(a), it can be seen that there is a separate layer of hypervisor running on top of a host operating system. This layer consists of a virtual machine monitor that provides full abstraction of virtual machines, which then act as a guest OS. In that case, all created VMs are isolated from each other and act as a separate OS on top of the physical host. Due to this hypervisor, every VM has its own kernel which makes the application more expensive. Moreover, all the binaries and libraries are also created along with a kernel, which is of course a great overhead in terms of resource allocation [7] [9]. Unlike any hypervisor-based system, there is another lightweight virtualization based on Linux containers. LXC removes that overhead and provide a full-fledge OS in the form of containers. It can be seen from Fig. 1(b). The kernel of host OS is shared between all containers, thus able to run many processes in an isolated fashion [7] [8].

There are many benefits of using container based virtualization which distinguishes it from the traditional hypervisor based virtualization [10]. Some of the benefits are described below:

A. Portability

Since LXC are portable so it can be run in any environment without changing the functionality. The applications running in a container can also be bundled together in a single unit and then deployed and separated into another environment.

B. Scalability

LXC can run on any Linux system and then scale up from one to hundred and then scale it down. Another interesting factor is the deployment from physical system into the cloud and then back to the system again. Thus, containers can be easily suited for a public cloud platforms with scale out applications.

C. Isolation

It provides a complete isolation between the processes running in a container. It also provides isolation within multiple containers. For that, it uses two separate kernel features namespaces and cgroups which will be discussed in section 3.

D. Flexibility

As there is no overhead while creating containers, so it has a flexibility to share multiple resources of the host operating system between different guests. Moreover, building, developing, and deploying new containers are easy, which in turn reduces time and providing good visibility.

Apart from all the benefits, there are two features which are still better in hypervisor based machines. These are security and isolation. Hypervisors provides more security as compared to containers because it has attack barriers functionality [5]. For isolation, the container-based virtualization is supposed to have a weaker isolation, as it works on OS-level and shares most of the resources of host operating system. However, from the user’s perspective, container-based virtualization looks like a stand-alone system with all the resources required for its functionality [7].

III. KERNEL FEATURES

In order to provide more isolation in container-based virtualization, there are six kernel features which LXC can use. These are: namespaces, cgroups, SELinux profiles, Seccomp policies, chroots, and kernel capabilities [6]. Two of them are the most important features which are discussed below.

A. Namespaces

The main purpose of using namespaces is to provide per process isolation within containers, and wraps a global system resource in an abstraction. It ensures that the processes are assigned to same namespace and run in only that space without interfering with other processes. It also ensures isolation within containers and guarantees that the containers only sees their own environment. Currently, there are six namespaces inside Linux implementation, which are described below [11] [12].

a) Mount namespaces (mnt): It allows the processes to have different view of the filesystem mount points. The processes which are in separate mount namespaces have their own filesystem layout different than the other processes. Thus providing the isolation to the set of filesystems.

b) UTS namespaces (uts): This namespace feature allows containers to have their own hostnames. It isolates two types of system identifiers; nodename and domainname, which are separate for every container.

c) PID namespaces (pid): It allows the processes to have different IDs by assigning different processing identifiers. Thus providing isolation on a number space. The processes which are in a different namespaces can have same IDs but hidden from each other. Process which is in parent namespace can still see other process in the child namespace.

d) IPC namespaces (ipc): This namespace offers isolation of the inter-process communication resources (such as POSIX message queues and System V IPC objects) between namespaces.

e) User namespaces (user): It allows processes to have different user and group IDs. Thus providing isolation outside and inside a user namespace.

f) Network namespaces (net): It provides isolation of networking resources and each container has its own network devices. Thus, each network space contains separate routing tables, iptables firewalls, and network interface controllers.

B. Cgroups (Control groups)

Control groups, known by cgroups, is the main kernel feature that allows the processes to limit and isolate resource usage such as CPU time, system memory, disk bandwidth, network bandwidth, and monitoring. LXC allocates these resources by dividing the application into multiple groups. Then the profile, based on these resources, are assigned to each
group. After that, the processes which are specific to that profile runs on a dedicated group. This provides isolation within the group and there is no interfering between multiple groups. Moreover, after creating groups, there is a way to monitor these groups along with denying or assigning further resources. It is also possible to reconfigure specific group dynamically on a running system. By creating the groups and assigning processes to it, hardware resource can be appropriated divided, thereby, increasing overall performance.

[12][13]

IV. DOCKER AS A RUNNING APPLICATION

This section will be described later.

V. CONCLUSION

In this paper, Linux containers (LXC) has been discussed along with the comparison between container-based and hypervisor based virtualization. The benefits of LXC are also illustrated which showed the importance of containers. Two important kernel features, namespaces and cgroups are also presented. These features are used to provide more isolation inside the container in order to make system as efficient as possible. Namespaces provides per process based isolation solution, while cgroups provide resource management solution in terms of dividing processes into groups. After that Docker platform is illustrated which is a running application of LXC and uses all the functionality of Linux containers.

REFERENCES