

CoreOS

RIBKA RUFAEL^{1, *}

¹ School of Informatics and Computing, Bloomington, IN 47408, U.S.A.

* Corresponding authors: rrufael@umail.iu.edu HID: S17-IO-3016

paper-001, February 25, 2017

CoreOS is a minimal Linux operating system that allows application to run on containers. CoreOS can be used on clouds from EC2, Rackspace, GCE or virtual machines such as Vagrant, VMware, OpenStack or on physical servers such as PXE, iPXE, ISO. CoreOS makes use of it's disk partitions to achieve continous updates to the operating system without manual interaction. In CoreOS service discovery is achieved by etcd, applications are run on Docker and process management is achieved by fleet.

© 2017 <https://creativecommons.org/licenses/>. The authors verify that the text is not plagiarized.

Keywords: CoreOS, Container, cloud, Linux

<https://github.com/cloudmesh/sp17-i524/raw/master/paper1/S17-IO-3016/report.pdf>

1. INTRODUCTION

CoreOS also known as CoreOS Container Linux is a light weight Linux operating system that is designed to be used for container infrastructure. CoreOS allows applications to run on containers so that there is abstraction layer between applications and the operating system. The separation of applications and operating system helps avoid dependencies. CoreOS can be run on clouds, virtual or physical servers. CoreOS allows the ability for automatic software updates in order to make sure containers in cluster are secure and reliable. It also makes managing large cluster environments easier. One of the differences between CoreOS Linux and traditional Linux distribution is that, in the case of traditional Linux distribution operating system, utilities (such as systemd) and software (such as Python and Java) are put together but in the case of CoreOS Linux only operating system and utilities are bundled together. In CoreOS Linux applications and software (such as Python and Java) are run on containers as shown in Figure 1 [1].

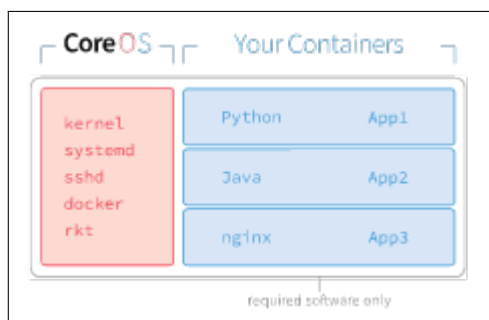


Fig. 1. CoreOS container layout [1].

CoreOS is an open source software that is provided by CoreOS, Inc. CoreOS, Inc also provides open source tools like etcd, rkt and flannel. CoreOS, Inc also has commercial products Kubernetes and CoreOS stack. These products can be used on top of CoreOS for additional functionalities like for example Kubernetes can be used for orchestration and management of containers that run on CoreOS [1].

Since CoreOS Linux is designed to be used in container frameworks, it makes it one of the candidates to be included into big data projects software stack that has container cluster infrastructure. Containers are multiple instances that contain applications and the softwares needed to run these applications sharing the same underlying operating system [2]. Big data projects in science or business involve processing of large amount of data. These big data projects are hosted on bare metal servers which are dedicated physical servers that are only used by one tenant or on clouds. Big data projects in science and in business sector can benefit from container frameworks that provide flexibility, ease of deployment, adding or removing container clusters based on demand [2].

2. COREOS ARCHITECTURE AND INSTALLATION

CoreOS has 9 disk partitions. CoreOS makes use of these partitions to achieve continous updates to the operating system. Having this disk layout aslo allows CoreOS to be able to reverse the updates in the event of unsuccessful updates. The disk partitions are: 1) EFI-SYSTEM which contains the bootloader and the partition type is VFAT. 2) BIOS-BOOT, ROOT-C and the 7th partition these partitions have no partition format and are reserved for future use. 3) USR-A and USR- B are the active and passive partitions that container linux sits on. Only one of them can be active at a time and partition type is EXT4 depending on which one is active. 4) OEM has EXT4 as partition type and this is where

OEM platform related configurations like custom networking and running an agent are stored. 5) OEM-CONFIG serves as an alternative place for an OEM. 6) ROOT may have EXT4, BTRFS, or XFS as partition type and it is used for keeping data which is persistent and this partition is stateful [1].

CoreOS Linux comes bundled with etcd, fleet and Docker. In CoreOS Container Linux service discovery is achieved by etcd, applications are run on Docker and process management is achieved by fleet. Service discovery in CoreOS Linux is achieved by etcd. Key value store on etcd is distributed and stored on all machines that run CoreOS Linux. State and configuration of each container running on CoreOS is stored in etcd key value store and this information is used to identify which container is selected to receive service requests. This service discovery capability makes it easy to add or remove machines. There is a command line interface that comes preinstalled on CoreOS linux called etcdctl that can be used to change and get key value data from etcd using etcdctl set and etcdctl get respectively. Another command that can be used for setting and reading key value is curl [3].

Container management in CoreOS linux is made possible by Docker. All applications run on Docker. Containers can be launched using Docker's command line interface [3].

the third component of CoreOS is fleet and it is used for management of containers with Docker installed. fleet is init system. init system is a process that gets started before any other process when a machine boots and runs till the machine is shutdown. fleet has fleetctl a command line interface that can be used to check the status of containers, to start containers and to stop containers [3, 4].

There is no package manager for installing, upgrading, configuring softwares in CoreOS. Softwares are installed as containers in CoreOS Linux operating system. In order to apply updates, CoreOS makes use of two root partitions one is active and the other is passive. First the updates to CoreOS are applied to passive partition and upon reboot all the updates are applied to the active partition [5].

CoreOS can be installed on clouds from EC2, Rackspace, GCE or virtual machines such as Vagrant, VMware, OpenStack or on physical servers such as PXE, iPXE, ISO [3]. Based on the information provided on CoreOS site, CoreOS container can be setup using Vagrant. Vagrant, virtual machine manager, can be used to run CoreOS on a single machine with Windows, OS X or Linux operating systems. It is recommended to have Vagrant version 1.6.3 or latest. Virtual machines by VirtualBox or VMware are both supported by Vagrant [6].

3. LICENSING

CoreOS Container Linux is an open source operating system. It is comprised of other programs and documents developed by other individuals and companies. All original components that are part of CoreOS Container Linux are licensed under Apache 2.0. At the time of writing is paper, the latest version of CoreOS Container Linux is 1325.1.0 [1].

4. PERFORMANCE

In a study done by Purdue University, performance tests between CoreOS 899.5.0 on Docker and Red Hat Enterprise Linux 6.5 were performed. In their study, the tests were run on 24-core AMD Opteron systems, with 2.1 GHz processors, 48 GB memory and 10Gbps Ethernet connection. For HPL performance tests,

the native RHEL 6.5 system with performance measurement of 7.839 GFLOPS performed slower than the container environment with CoreOS on Docker with performance measurement of 7.811 GFLOPS. For the network throughput measurement iperf was used. The results for RHEL 6.5 system were 8.26 Gbps for upload throughput and 9.38 Gbps for download throughput. The result for container environment with CoreOS on Docker were 8.43 Gbps for upload throughput and 9.37 Gbps for download throughput. Network File System throughput was done using iofzone and results are shown in Figure 2 [2].

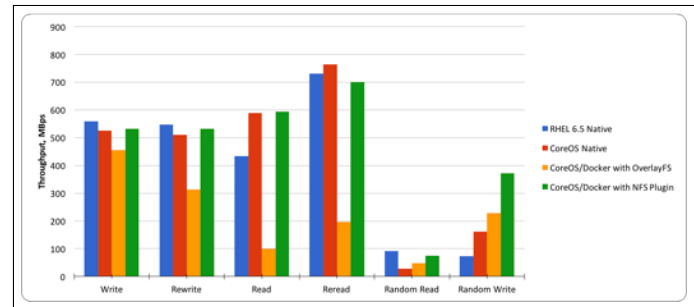


Fig. 2. File Server throughput [2].

In another study done in Institute of Informatics – Federal University of Rio Grande do Sul, performance analysis of ClickOS, CoreOS and OS^v was performed. Network Functions Virtualization(NFV) allows virtualization of network services such as firewalls and DNS. In their paper comparison of ClickOS, CoreOS and OS^v as NFV based tools was performed. They compared boot time, response time and memory consumption for the three virtualization technologies. As per their evaluation result, ClickOS has the lowest boot time and response time followed by CoreOS and with regards to memory consumption, CoreOS has the smallest memory usage followed by ClickOS. Based on the result OS^v has the slowest boot time and response time as compared to CoreOS and ClickOS. Also for memory consumption, OS^v has the highest consumption as compared to the other two technologies used [7].

5. USE CASES

In this section use cases from big data and other areas where CoreOS Linux is used are discussed.

5.1. Use Cases for Big Data

Metagenomics RAST server(MG-RAST) is free access portal that can be used by researchers who study microbes for accessing and analyzing metagenomics data(Random community genomes) [8]. CoreOS Container Linux and fleet are the technologies used on MG-RAST container application servers. Researchers can input data into MG-RAST portal through script, web site or REST API [9].

5.2. Other Use Cases

CoreOS Container Linux on Amazon EC2 cloud was used to implement the project Two-stage Stochastic Programming Resource Allocator (2SPRA). The language used to implement was Python. In this project, they tried to address the problem that exist in most data centers of over provisioning resources in order to achieve performance service level objective. 2SPRA is a resource allocation scheme and it is able to optimize resource allocation

for containerized web services based on varying workloads. 2SPRA analyzes the relationship between change in workload, resource allocation and response latency in order to calculate the the number of containers needed. In this experimental work, CoreOS was used in order to simulate real word scenarios of n tier application servers running on containers. The test architecture has client Java based emulator which creates multiple user sessions at the same time, web hosting platform with where RUBis benchmark is installed on Virtual machine with CoreOS version stable r717.3.0 and the third component is 2SPRA implemented in Python running on Virtual machine [10].

6. EDUCATIONAL MATERIAL

CoreOS website contains detailed documentations and materials for anyone who is interested in setting up CoreOS Container Linux on clouds, virtual or physical servers [3]. Users who are interested can contribute to the CoreOS open source projects through GitHub [11].

The book titled CoreOS Essentials, also has information about CoreOS overview and its installation [5].

7. CONCLUSION

CoreOS Linux is a light weight Linux based operating system that is designed for containers. It provides abstraction by separating the operating system from application and softwares. Operating system updates to CoreOS are automatic without a need for user interaction. CoreOS can be installed on clouds from EC2, Rackspace, GCE or virtual machines such as Vagrant, VMware, OpenStack or on physical servers such as PXE, iPXE, ISO.

CoreOS Linux aims to make applications and microservices running on containers secure, easy to deploy and portable. Big data projects can leverage these benefits that come with CoreOS Linux by adding it to their infrastructure.

As previous performance results have shown, CoreOS Linux has the lowest memory usage compared to other operating systems namely ClickOS and OS^v [7].

ACKNOWLEDGEMENTS

The author would like to thank Professor Gregor von Laszewski and associate instructors for their help and guidance.

REFERENCES

- [1] CoreOS, "Why CoreOS," Web Page, Jan. 2017, accessed: 2017-01-23. [Online]. Available: <https://coreos.com/why/>
- [2] S. Julian, M. Shuey, and S. Cook, "Containers in Research: Initial Experiences with Lightweight Infrastructure," in *Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale*, ser. XSEDE16. New York, NY, USA: ACM, 2016, pp. 25:1–25:6. [Online]. Available: <http://doi.acm.org.proxyiub.uits.iu.edu/10.1145/2949550.2949562>
- [3] CoreOS, "CoreOS Quick Start," Web Page, Feb. 2017, accessed: 2017-02-17. [Online]. Available: <https://coreos.com/os/docs/latest/quickstart.html>
- [4] Wikipedia, "init," Web Page, Mar. 2017, accessed: 2017-03-28. [Online]. Available: <https://en.wikipedia.org/wiki/Init>
- [5] R. Mocevicius, *CoreOS Essentials*. Packt Publishing Ltd, Jun. 2015.
- [6] CoreOS, "Vagrant," Web Page, Feb. 2017, accessed: 2017-02-17. [Online]. Available: <https://coreos.com/os/docs/latest/booting-on-vagrant.html>
- [7] L. Bondan, C. R. P. dos Santos, and L. Z. Granville, "Comparing virtualization solutions for NFV deployment: A network management perspective," in *2016 IEEE Symposium on Computers and Communication (ISCC)*, Jun. 2016, pp. 669–674.
- [8] F. Meyer, D. Paarmann, M. D'Souza, R. Olson, E. M. Glass, M. Kubal, T. Paczian, A. Rodriguez, R. Stevens, A. Wilke *et al.*, "The metagenomics RAST server—a public resource for the automatic phylogenetic and functional analysis of metagenomes," *BMC Bioinformatics*, vol. 9, no. 1, p. 386, 2008. [Online]. Available: <http://dx.doi.org/10.1186/1471-2105-9-386>
- [9] A. Wilke, J. Bischof, W. Gerlach, E. Glass, T. Harrison, K. P. Keegan, T. Paczian, W. L. Trimble, S. Bagchi, A. Grama, S. Chaterji, and F. Meyer, "The MG-RAST metagenomics database and portal in 2015," vol. 44, no. D1, pp. D590–D594, 2016. [Online]. Available: <http://dx.doi.org/10.1093/nar/gkv1322>
- [10] O. Adam, Y. C. Lee, and A. Zomaya, "Stochastic Resource Provisioning for Containerized Multi-Tier Web Services in Clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. PP, no. 99, pp. 1–1, 2016.
- [11] CoreOS, "CoreOS," Web Page, Jan. 2017, accessed: 2017-02-26. [Online]. Available: <https://github.com/coreos/>