Docker container based analytics at IoT edge

Video analytics usecase

Pankaj Mendki
Senior Principal Engineer, Member of R&D,
Talentica Software Pvt. Ltd
Pune, India
pankajm@talentica.com

The number of internet connected devices is growing rapidly. It is getting difficult to gather and process the huge volume of the data generated from these devices. Edge computing paradigm is evolving that helps to process the data locally close to the source. This paper highlights using docker container based analytics services at the edge for data processing. Feasibility is studied for setting up deep learning framework on Raspberry Pi for real-time analysis of video feed from the surveillance camera. Performance benchmarking is also done to figure out overhead of containerization.

Keywords—IoT; edge computing; docker; video anaytics; mxnet; surveillance camera; raspberry pi;

I. INTRODUCTION

Adoption of the internet of things (IoT) technology is gradually increasing. It is predicted that the number of internet connected devices will reach to about 50 billion value by 2020 [1]. A typical IoT stack consists of sensors with constrained resources sending data to device that has internet connectivity – this device is called gateway. Gateway sends the data to servers for further processing over internet. Generally these servers are cloud based. With increasing number of sensors, total number of data points and volume of data can grow very high, e.g. a single turbine blade can generate 500GB data per day [2]. Processing this high volume of data demands high bandwidth, high computation resources at the cloud and it also results in higher latency.

Fog computing [4] and edge computing are the new paradigms that will help reducing the compute and storage resource overloading at cloud infrastructure by exploiting the resources close to the data source (sensors). This paper discusses simplifying application deployment at the IoT edge using docker - lightweight virtualization mechanism. This paper highlights benefits of using docker at the edge, like ease of deployment, upgrade and simplicity in scaling horizontally.

This paper's objectives are to evaluate possibility of modularizing the analytics operations using container virtualization. This will enable easy movement of that operation to network edge from cloud and scaling the operation horizontally at edge over heterogeneous environment and also at the cloud if required. Another objective is to evaluate performance overhead because of the container virtualization at the edge for a typical analytics operation.

This paper is organized as follows: section II describes high level concepts of edge computing and docker, section III describes implementation requirement for edge computing application, section IV has lists down a typical use case for edge computing application, section V illustrates different components used in the implementation, section VI is about the performance benchmarking for the docker based solution and section VII is the conclusion section.

II. BACKGROUND

This section provides high level overview about the edge computing and the container technology as enabler for the edge computing.

A. Edge Computing

Edge computing is one of the evolving areas in the field of IoT. In most of the IoT based deployments, multiple sensors send data to a gateway and the data are forwarded to cloud based infrastructure. All the processing, computation and storage happen at the cloud side. As the number of IoT enabled sensors are likely to grow, this is going to exponentially increase the data size send to the cloud. This will increase data transfer cost, data processing cost and in turn overall operational cost for the solution. Processing data on the cloud also adds to the data processing latency. Factors contributing to this latency are network transfer and slowness because of server overload. The idea behind edge computing is exploit edge resources for compute and/or storage. Edge computing is not a substitute for the cloud but rather it's a complementary to the cloud based processing. Typically edge level devices are single board computer (SBC) equipped with an operating system.

B. Operating system level virtualization

Hypervisor based virtualization has evolved over a decade and it has been widely getting used. The host machine where hypervisor is installed can run multiple virtual machines. Each virtual machine (VM) in turn can run one or more applications. Every virtual machine has entire operating system installed which could be same or different from that of the host machine. Virtual machines can be migrated from one host to another.

Operating system (OS) level virtualization also known as container based virtualization. Containers use process isolation

978-1-5090-6785-5/18/$31.00 © 2018 by IEEE

at the operating system level. Typically one application is deployed inside a container. Figure 1 depicts the container based application architecture. They share the kernel of the host machine operating system. Containers are lightweight compared to VMs in terms of image size, boot-up time and resource consumptions. It is possible to quickly start new container or migrate existing container within a few milliseconds.

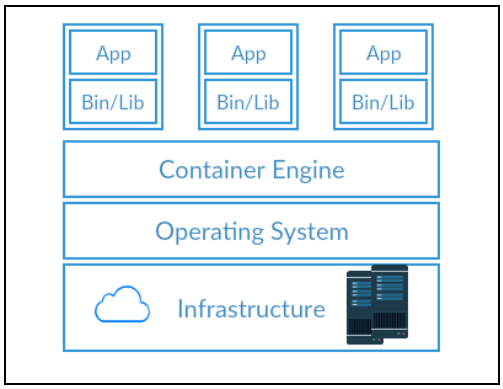

Fig. 1. Docker Application Architecture

In this paper [5], it is observed that containers performance is better than that of hypervisor based systems especially in Disk I/O based operations. Container based virtualization being lightweight facilitates more dense deployment of services.

There are multiple container based solutions available; Docker is used as container platform for the work in this paper.

III. Implementation Requirements

Some of the recent works [6], [7] have already evaluated docker based containerized environment for edge computing. Studies in [7] have concluded docker to be suitable for edge computing based on these criteria: (i) Deployment and termination of services, (ii) Resource and service management, (iii) Fault tolerance, and (iv) Caching capabilities. This work [8] has explored using docker for supporting multi-tenancy at the gateways and proposed the paradigm gateway as a service.

High level requirements for implementing data analytics solution at the edge can be summarized as follows

A. Ease of deployment in heterogeneous environment

The functionality that is to be deployed to the edge may co-exist in cloud and / or edge depending upon the load. It is also possible to have multiple installations of the same functionality. There is a lot of heterogeneity in the edge devices in terms of operating systems, CPU architecture, communication mechanism etc. Docker supports multiple CPU architectures including ARM.

B. Ease of service and device orchestration

The edge level functionality can be considered as a standalone service. It is possible to have multiple such services based upon the sensor data processing and applications. There are multiple tools available for orchestrating these services. Docker provides built-in swarm mode [9] for service and cluster orchestration. This facilitates functionality like scaling the service, service discovery, load balancing, high availability of service and updating the services.

Docker images can be cached at the edge in local registry. Thereby pushing the updates at the edge, newer version of services can be quickly deployed.

C. Isolation and multi-tenancy

Every docker container runs one instance of a service and container is running in an isolation mode i.e. independent of other containers [note – network traffic between container and host is not fully isolated]. This is an enabler to achieve the multi-tenancy at the edge. One edge device may house multiple edge services.

IV. Usecase

This section describes use case about video analytics at the edge.

Surveillance video cameras are getting used widely for the security purpose. CCTV survey in U.K. estimated that there is one surveillance camera per 11 people in UK [10]. Most of the times, these cameras are used as postmortem tool to analyze the situation after the incident has happened. Use of edge level analytics can help to preventively diagnose the situation and raise alerts before the incident.

The idea of analyzing the video feed using video processing framework is not new. But generally this analysis is not real time. It is done offline on the stored video feed. The use case here is to analyze the surveillance camera video feed in real time close to the source using single board computer device. It should raise alert in case an incident is about to occur. A typical incident could be detecting a plausible threat situation. Number of deep learning frameworks are available that can analyze the video feed and detect threat situations. The challenge is to make them run real time on the single board computers. Another challenge is that one board may need to analyze feeds from multiple cameras.

V. Implementation

This section describes about the different components used for the edge computing solution for the above use case.

978-1-5090-6785-5/18/$31.00 © 2018 by IEEE

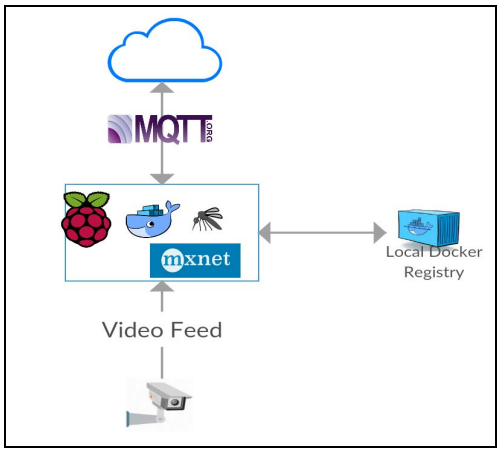

Fig. 2. Setup Components

Raspberry Pi 3 Model B is used as gateway. Raspbian Jessie Lite operating system is installed on the board. Docker engine version 17.06 is installed. Raspberry Pi is configured to receive video feed using a camera module. Video feed is configured to be 1080p (1920x1280) at 10 frames per second. A typical surveillance camera in public place can provide good surveillance at 5 to 7.5 fps [11].

As a first step, a proof of concept is implemented to benchmark how many frames per second can be processed by Raspberry Pi board. Input used is a video file with 1080p resolution and 10 fps. Multiple deep learning frameworks are available that can analyze the video feed. This paper uses mxnet [12] framework for analysis. This framework supports ARM architecture and hence can be installed on Raspberry Pi in a containerized environment. A pre-trained deep residual network model Renet-152 [13] is used for the analysis. This model is loaded in memory one time when the program starts and every frame is analyzed for the standard thread scenarios. MQTT client is used to send the analyzed information including alert to the cloud based server. It is possible to run multiple containers each processing different frame making the system doing parallel processing. Docker swarm is used to orchestrate this service and it can deploy the service across multiple Raspberry Pi instances over the network there by scaling the service horizontally at the edge.

Local docker registry is setup close to edge; services can be updated by updating the local registry docker image. Figure 2 depicts the setup components.

VI. Benchmarking

This section analyses if there is any performance overhead of using containers over bare metal Raspberry Pi. Similar analysis is done in the paper [6], [7] concluding the minimal impact for CPU, Disk I/O and memory intensive processes, whereas network intensive process has some impact of using container.

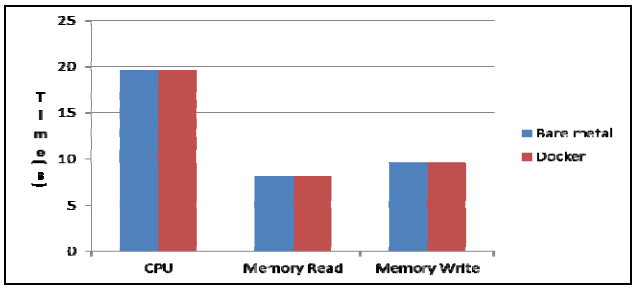

Fig. 3. CPU and Memory Benchmarking Results

Figure 3 shows that CPU and memory performance results are almost identical on docker and bare metal. To get these results, sysbench tool is used.

Figure 4 shows performance comparison for sequential disk I/O throughput and TCP based network I/O throughput. For docker, throughput is slightly lower as compared with the bare metal. Fio is used for disk I/O performance evaluations are and for network I/O iperf3 is used.

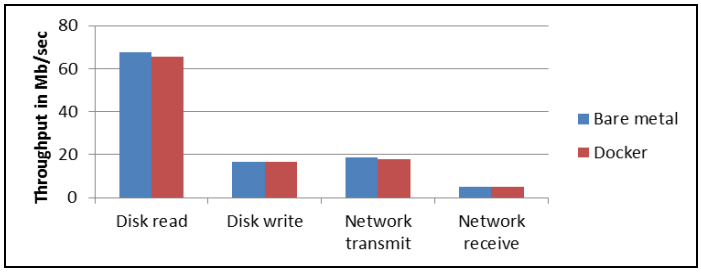

Fig. 4. Disk and Network I/O Performance

Mxnet performance results with and without docker, are almost identical showing docker has minimal impact on the performance when used on top of bare metal. Figure 5 shows comparison in the average load time for resnet-152 model and average analysis time per image file for set of 10 files with 1080p resolution.

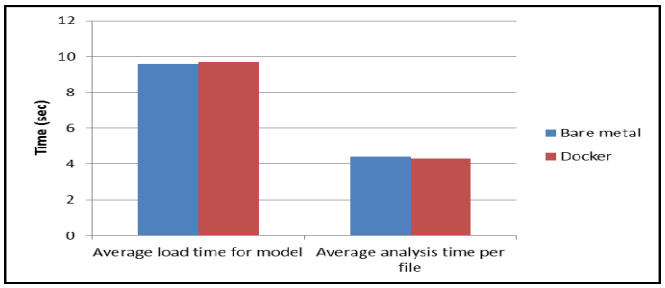

Fig. 5. Mxnet Performance

Average mxnet processing time for a single 1080p image is about 4-5 milliseconds. It is possible to process a video feed at the rate of 10 fps using a single container.

978-1-5090-6785-5/18/$31.00 © 2018 by IEEE

VII. Conclusions and Future Work

Container based virtualization is a good choice for deploying edge level analytics solution. Deep learning framework can be successfully deployed inside a docker container on single board computers like Raspberry Pi for analyzing the video feed in real-time. This solution has negligible overhead in terms of CPU processing compared with the bare metal deployment. Deploying the analytics solution in docker container at the edge provides ease of service management and orchestration.

References

[1] "Internet Of Things Will Deliver $1.9 Trillion Boost To Supply Chain And Logistics Operations," Internet Of Things Will Deliver $1.9 Trillion Boost To Supply Chain And Logistics Operations | The Network | The Network. [Online]. Available: https://newsroom.cisco.com/press-release-content?articleId=1621819.

[2] S. Kattau, "Research from Gartner: Real-Time Analytics with the Internet of Things", RTInsights, 2018. [Online]. Available: https://www.rtinsights.com/research-from-gartner-real-time-analytics-with-the-internet-of-things-dw/

[3] O. Salman, I. Elhajj, A. Kayssi, and A. Chehab, "Edge computing enabling the Internet of Things," 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), 2015, pp. 603-608..

[4] F. Bonomi, M. Rodolf, Z. Jiang, and A. Sateesh, "Fog computing and its role in the internet of things." In Proceedings of the first edition of the MCC workshop on Mobile cloud computing, ACM, 2012, pp. 13-16.

[5] R. Morabito, J. Kjallman, and M. Komu, "Hypervisors vs. Lightweight Virtualization: A Performance Comparison," 2015 IEEE International Conference on Cloud Engineering, 2015, pp. 386-393.

[6] R. Morabito and N. Beijar, "Enabling Data Processing at the Network Edge through Lightweight Virtualization Technologies," 2016 IEEE International Conference on Sensing, Communication and Networking (SECON Workshops), 2016, pp. 1-6.

[7] B. I. Ismail, E. M. Goortani, M. B. A. Karim, W. M. Tat, S. Setapa, J. Y. Luke, and O. H. Hoe, "Evaluation of Docker as Edge computing platform," 2015 IEEE Conference on Open Systems (ICOS), 2015, pp. 130-135.

[8] R. Morabito, R. Petrolo, V. Loscri, and N. Mitton, "Enabling a lightweight Edge Gateway-as-a-Service for the Internet of Things," 2016 7th International Conference on the Network of the Future (NOF), 2016, pp 1-5.

[9] "Swarm mode overview," Docker Documentation, [Online]. Available: https://docs.docker.com/engine/swarm/.

[10] D. Barrett, "One surveillance camera for every 11 people in Britain, says CCTV survey," The Telegraph, 10-Jul-2013. [Online]. Available: http://www.telegraph.co.uk/technology/10172298/One-surveillance-camera-for-every-11-people-in-Britain-says-CCTV-survey.html.

[11] "School District Standardization of CCTV." [Online]. Available: https://www.sde.idaho.gov/student-engagement/sdfs/files/school-safety/Security-Camera-Infrastructure.pdf

[12] "MXNet: A Scalable Deep Learning Framework," MXNet: A Scalable Deep Learning Framework. [Online]. Available: http://mxnet.io/

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778.

978-1-5090-6785-5/18/$31.00 © 2018 by IEEE