CS-C3140 Operating Systems
Challenge exercise - phase 1

28. syyskuuta 2017

Authors

Oliver, Struckmeier, 664064, oliver.struckmeier@aalto.fi
Nikita, Korhonen, 659150, nikita.korhonen@aalto.fi
Paula, Minni, 62686F, paula.minni@gmail.com

Technical report - Scalability, Execution and Communication Sandboxing

This report is written in the pre-study phase of the challenge exercises in the lecture CS-C3140 Operating Systems. It covers basic definitions of technologies and information about the given tools. Furthermore it will be discussed how the tools and technologies can be used to achieve scalability, execution and communication sandboxing.

1 Basic Definitions and introduction of the tools

Sandboxing describes the process of running programs in an environment that is in some way separated from the host hachine or its operating system. Instead of using the systems resources the program is run in the "sandbox"which is an environment for the program that contains a controlled set of resources needed to run the program.

1.1 Microservices, Containers vs. virtual machines

1.1.1 Containers and Microservices

Microservices is architectural concept for implementing an overall service through splitting its components in different processes. These processes can be located on different devices working on different operating systems and they communicate between each other through external apis. In addition the microservices are to be considered independent from each other. This means that the application promotes modularity and allows parallelized development as well as independet scaling of the respective services. Containers are an abstraction at the app layer that packages code and dependencies together. Multiple containers can run on the same machine and share the OS kernel with other containers, each running as isolated processes in user space. Therefore, containers are a way to implement microservices on a single machine. [2, 3, 4, 5]

1.1.2 Virtual Machines vs. Containers

Compared to virtual machines containers are using the resources of the OS using an API instead of relying on a hypervisor which include a fully embedded OS. This means that a container has discrete components of application logic and therefore provides only the minimum of resources needed to do their job. As a result containers are more efficient and require less resources than virtual machines and are therefore more lightweight, easily packed and efficient for large scale systems. Note that there are application containers and system containers. Application containers run only single service, while system containers have several services running on containers operating system. Drawback of containers compared to VMs is that for example Windows container cannot run on a Linux machine or vice versa.

1.2 Execution Sandboxing in Image Processing

1.2.1 Python's scikit package and scikit-image + scikit-learn

Scipy is a bundle of open-source software for mathematics, science and engineering containing tools for advanced mathematical computation as well as graphical display and data analysis.

An important collection of tools for more specific purposes are the Scikits. In this scope the most important ones are scikit-image and scikit learn. Scikit-image is a SciPy compatible python package with algorithms for image

processing with SciPy, including I/O, morphology, filtering, color manipulation and object detection.

Scikit-learn is another SciPy compatible package used for machine learning applications and includes tools for data mining and data analysis. It can be used to classify objects using image recognition and clustering similar objects into sets. It's main functionalities are various classification, regression and clustering algorithms.

These packages can be used to process data from different input sources like sensors. This also means that it is possible to scale applications relying on this kind of information by packing them in containers. [6,]

1.2.2 OpenCV

OpenCV is a library for real-time computer vision. It provides interfaces for C++, C, Python and Java. It is OpenCL enabled and can take advantage of hardware acceleration provided by the platform it is running on. In the scope of this report this means, that OpenCV could run for example in one or more containers or a virtual machines to handle input form one or more cameras. [7]

1.3 Communication Sandboxing with Machine Learning and Processing Pipelines

Pipelines are a group of data processing units that are connected in series where data is passed through an unidirectional stream from one unit to the next one. The data stream flowing through a pipeline consists of records, bytes or bits. Therefore, communication sandboxing can be achieved using certain tools like TensorFlow or Apache Storm and describes the process of using processing pipelines to handle data in a separate environment or distributed computation system. This data can be image data from for example OpenCV or other sensor data acquired by a systems sensors. The input data streams can be processed and redirected in any way to get the results needed.

1.3.1 Tensor Flow, Nolearn and Apache Storm

A few examples for applications used to work with data streams are Tensor Flow, Nolearn and Apache Storm. Tensor flow for example uses flow graphs to process data transfered through the edges with nodes in the graph being mathematical operations. Its architecture is flexible and allows deployment on one or more devices with a single API. Again this shows that Tensor Flow can be run in containers or virtual machines to work across multiple machines

or computation environments. The Nolearn library is similar to this and provides abstraction wrappers around existing neural network libraries. Apache Storm is a distributed realtime computation system that can process streams of data in real-time while providing interfaces to different programming languages as well as the possibility to use containers to scale the services. [8, 9, 10]

1.4 Results

In conclusion, microservices architecture and containers provide the scalability, modularity and security for a system. Communication sandboxing, achieved with tools like Tensor Flow and Apache Storm, provides modularity for execution of image processing.

Finally it can be said that all the above mentioned systems and libraries serve the purpose of scaling the flow of information from one or more sensors. The acquired data can be processed by using multiple systems with different operating systems or hardware structures.

2 References

1. mycourses, https://mycourses.aalto.fi
2. Computerweekly - What are containers and microservices?, http://www.computerweekly.com are-containers-and-microservices
3. Wikipedia Microservices, https://en.wikipedia.org/wiki/Microservices
4. Docker, https://www.docker.com/what-container
5. lxd, http://www.ubuntu.com/cloud/lxd
6. scikit-image, http://scikit-image.org
7. OpenCV, http://opencv.org
8. nolearn, https://github.com/dnouri/nolearn
9. Apache Storm, storm.apache.org/
10. Tensor Flow, https://www.tensorflow.org/
11. Operating System Containers vs Application Containers, https://blog.risingstack.com/opera system-containers-vs-application-containers/