Seminar

USING LIGHTWEIGHT VIRTUALIZATION THREE TIER ARCHITECTURE TO STREAMLINE IOT APPS DEVELOPMENT

Department of Computer Science
Kristianstad University
Course: DT584C
Master's in computer science

Student: Hazrat Ali <react.dev.se@gmail.com>
Teacher: Qinghua Wang

Date: 12 January 2020

Table of contents

Abstract

IoT devices have limited resources and need considerable amount of time to be integrated with software applications. There is usually more time spent in configuring these devices than actually developing IoT applications. Therefore we devised the use of lightweight virtualization technology(containers) as a middle tier to simulate IoT devices in our three tier architecture of software application development .

The benefit is that It gives flexibility in managing and distributing software, cross platform capabilities. The build process is faster, large number of services and their quick deployments in containers.

Also leveraging some IoT functions and tasks to cloud eg storage, image processing. We also proposed a standard API for IoT services.

In the future this approach, which will enable scalability, security and manageability and also remote access.

Introduction

The integration of IoT and cloud services opens a plethora of doors in delivering services. This phenomena of technology growth integrates more businesses from private, public sectors resulting better technology integrations.

Like many other technologies there are several challenges that hinders the development of IoT applications and services. In this article we figure out those challenges and provides solutions using Virtualization technologies.

Resource virtualization is a well known and widely used concept in technological world. There are several technologies available depending on the scenarios and use cases.

There are five major categories in which virtualization can play an important role, ie Application, Desktop, Hardware, Network and Storage.

The widely used virtualization techniques are virtual machine, KVM, QEMU, VMWare, virtual pc, Xen, Virtual box. Hypervisor requires a virtual machine software that runs on top of hosting operating system and provides a full abstraction of physical machine resources. It gives full virtualization environment but at the same time its heavy and requires enough hardware resources to be run on hosting systems. However recently a lightweight virtualization alternative was introduced which is known as container based virtualization[1].

Container based virtualization (aka OS level virtualization) partitions the physical machine resources into multiple isolated instances. Since it is lightweight and requires less resources it could further be tailored for IoT applications and virtualization needs.

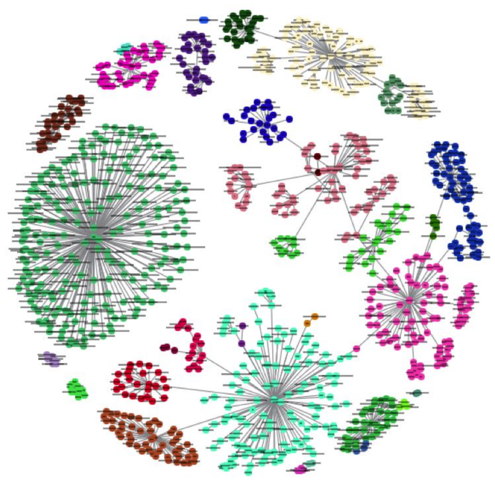


Fig 1 - Typical internet of things network

Container based virtualization such as docker, OpenVZ are lightweight alternative that can be adopted on smart IoT objects for enhancing the IoT cloud services[1]. The lightweight feature embedded in containers make it useable for IoT devices integration as there might be a large number devices is some scenarios.

Virtualization in connection with cloud services also helps in deploying pieces of software on smart devices in a very flexible fashion[1].

Motivation

IoT has revolutionized several aspects of our life like smart homes, smart buildings etc. Using IoT a heterogenous amount of devices are connected in IoT devices leading into interoperability[2].

To solve these issues several alternative research articles has addressed it. A gateway might be used as a service to enable such interoperability using virtualization technology[2]. It has also shown that it provides best results in terms of performance.

Lightweight virtualization techniques[3] have been used to support and streamline IoT services and applications. In a modern IoT application like in the scenario of smart homes, automated vehicles etc we need miliard of services and devices. Implementing these are not always ideal or cost effective for rapid and flexible development.

Since a lot of IoT applications are evolved with many challenges and services, therefore cloud services suits best in these scenarios. This could be witnessed in past by the emergence of many cloud services from bigger companies like google cloud, firebase, amazon web services and IFTTT. The use of these services and even making smart portable libraries and services on top of them improve the application development process even further. Providing standards and a common API for these devices is challenging but could be leveraged with the use of virtualization.

Another requirement for virtualization is the need of services that are sensitive to latency and bandwidth[3]. Therefore virtualized containers can help in improving this drawback in centralized computing. Although this approach is not suggesting replacing the cloud but instead empower service development in some special use cases.

A very close example is edge offloading which identified the gap between IoT devices and cloud and suggested an intermediate unit to augment and enrich this interaction[4].

By harnessing the power of distributed edge resources we can support novel service scenarios such as autonomous vehicles, smart cities, and augmented reality[3].

Methodology

Keeping in view the studies we attempt to find the answer to the question.

Q1. Can we enhance IoT apps development using virtualized container based IoT devices ?

Q2. Can we suggest a common API standard for IoT devices to access a function ?

For example if an app used devices like Nordic Thingy 52, HS100 or similar and that could be made available using a simulator which also provide a transparent API structure as the original devices. This way the apps developer will only have to focus on the business logic and development process of the app itself instead of hardware and software issues with smart devices. This idea can be extended to provide remote access to containerized virtual devices by universities to students remotely. This will enable students and researcher to even access a suit of IoT devices remotely. Virtualization provides a flexibility of managing and scaling these devices for any experimental needs.

- Identify with feature of the device can be easily replicated using node and a containerized docker image
- Study its API documentation and implementation details with device
- Find the metrics and range of their values
- Replicate and test
- Repeat for other desired features and devices

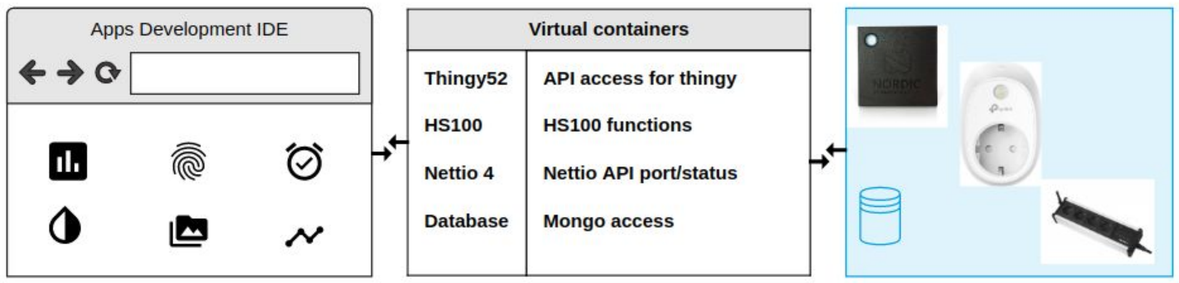
Implementation

The architecture is composed of three major components: IoT Application development suit, IoT Virtualized environment (with cloud access) and IoT devices. The placement of middle layer between end devices and Apps/Services is an architectural decision and is widely used in other implementations[,5,6] as well such as Network function virtualization, firewall, network address translators, and virtualization of switches etc.

The virtualize component aims to provide access to cloud services, which is a good place for catching and even further enhancing services. This is the main place and focus of this article, and therefore it should entail programmability and flexibility to add and test both device virtualization and other cloud services required by any IoT application (mobile, desktop or web) and service. The cloud service could be cached or localized[3] here, however we are not aiming it in this article but giving provision for it.

Setup

- Operating Systems Ubuntu
- Docker
- Install Java
- Configure firebase
- MongoDB
- Node and npm



(a) (b) (c)

Fig 02: Lightweight technology 3 tiers architecture

a. The application source code IDE

b. Virtual environment - Simulator in Java

c. Smart IoT devices connected with simulator

From architectural point of view it is obvious that the middle layer which contains the virtual docker containers should be flexible enough and provide easy to use API. Since it uses API to provide assistant to application development layer it can be ideally implemented in any programming language and development toolkits. This middle layers containers could be managed by providing a small and easy to use Java based simulators.

The core functionalities of the simulator would be add, remove, connect devices and show responses or visualize the state of IoT devices. At its functionality level it may introduce further enhancement of virtualized environment in terms introducing noise or packet loss of bluetooth, wifi mediums. This will help and produce different test cases for application testing later.

The layer (c) of Fig 2 above depicts the real sensor devices that would be first connected by the simulator but if they are not accessible or out of service the virtual container based device will spin up and start working automatically providing a transparent software development experience. Raspberry PI is a single board computer which finds its way in IoT devices context especially for providing gateway functionality[7].

In our previous work [8] we have used raspberry PI as a gateway and also as a web server for serving IoT web applications. Our three tier architecture we can replicate the same functionalities such as mango, firebase, web server etc in our Simulator using docker containers.

Docker containers

Docker container provides a different level of sirtualizatin abstractions which is light weight as it in replicating hosting services but instead abstracting them. This feature makes very light weight and we can spin a lot of virtual instances with using too much system resources. Unlike hypervisors there is no separate operating systems with its own ram, hard disk and other resources. This can be depicted in the figure (source: docker.com) below.

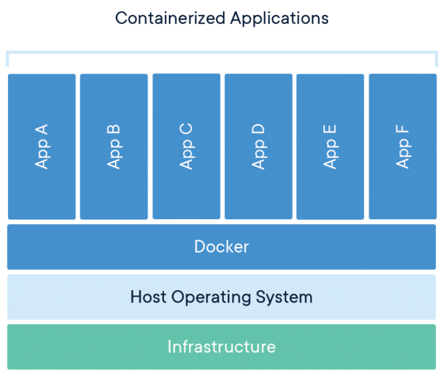


Fig: 03 - docker containers

Docker provides an underline container engine with an API that helps in managing, adding and removal of containers. It can encapsulate a whole application along with its dependencies within a virtual container[1] which has lower overhead on host operating system.

There are two general usage models of containers[1]:

- **Application container :** in this container a single application runs
- **System container :** this container has an instance of user space and allows for multiple isolated instances of user space to run at the same time each one with its own Init process, file system and network stack.

Docker provides a very flexible setup, configuration and management of containers with functioning APIs, therefore we have chosen containers approach for virtualization in our Java based simulator.

Java simulator

Java is a famous and cross platform programming language enriched with object oriented principles. It makes not only secure but also provides a lot of useful libraries out of the box. The main functional points identified are stated below.

- A graphical workspace with necessary menus, toolbar developed in Java FX.
- A graphical toolbox will be developed with drag and drop features. This will contain the different types of IoT devices, and as soon as it is dropped in the workplace, a new corresponding docker container will spin down as shown in Fig 2.b.

- The docker instance will provide RestAPI endpoint url which will implement all desired features of the corresponding IoT device.
- The necessary buttons to start / stop (if needed) should be developed.
- To enhance further we can provide a common restful API paradigm for all devices. Such as
    i. `/api/v1/status/1`
    ii. `/api/v1/status/0`
    iii. `/api/v1/radio/1`
    iv. `/api/v1/radio/0`
    v. `/api/v1/gpio/1/0`
    vi. `/api/v1/gpio/1/1`

All of the above is in the format of api/v1 which is the version number of the api, status/radio shows the feature of the corresponding IoT devices(hs100, thingy, raspberry PI).

Discussion

- Virtualization is a very flexible and affordable technology. After reviewing several articles we found that container based virtualization is more suitable in case to use less resources. This technique could is termed is lightweight virtualization as it is not resource hungry.

- Many virtual instance can be spin around as per the need of the IoT application under development. To manage all these services we propose a Java based simulator to help do the most common operations on containers.

- The configure can be made simpler, once configured an instance it could be easily replicated or even reconfigured using programming APIs of docker.

- In addition to leveraging the capabilities of docker for enhancing rapid application development of IoT applications it can also be installed cross platform because of the use of Java virtual machines.

- Moreover, there are fundamental hardware requirements to run such services on IoT devices because of their scarce resources for example mongodb or image processing capabilities. However our three tier architecture gives the possibility to run such services in the middle tier.

Conclusion

- Shifting our focus on IoT Apps development, we identify that IoT devices could be segregated from the software development process by using lightweight docker containers. To achieve this goal the challenge of transparency is achieved by using the same API structure as of the real IoT device.
- In fact we can also improve a step further by using standard Restful API for all IoT devices that could address the challenges of using different API structures.
- The lightweight implementation of docker containers gives the flexibility and eas of configuration for rapid IoT applications development. At the same time it's not only an affordable solution but we propose a remote access to it in the future studies.

References

[1] A. Celesti, D. Mulfari, M. Fazio, M. Villari, and A. Puliafito, “Exploring Container Virtualization in IoT Clouds,” in *2016 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2016, pp. 1–6, doi: 10.1109/SMARTCOMP.2016.7501691.

[2] R. Morabito, R. Petrolo, V. Loscrí, and N. Mitton, “Enabling a lightweight Edge Gateway-as-a-Service for the Internet of Things,” in *2016 7th International Conference on the Network of the Future (NOF)*, 2016, pp. 1–5, doi: 10.1109/NOF.2016.7810110.

[3] R. Morabito, V. Cozzolino, A. Y. Ding, N. Beijar, and J. Ott, “Consolidate IoT Edge Computing with Lightweight Virtualization,” *IEEE Netw.*, vol. 32, no. 1, pp. 102–111, Jan. 2018, doi: 10.1109/MNET.2018.1700175.

[4] V. Cozzolino, A. Y. Ding, and J. Ott, “Fades: Fine-grained edge offloading with unikernels,” in Proceedings of the Workshop on Hot Topics in Container Networking and Networked Systems,ser.HotConNet ’17. New York, NY, USA: ACM, 2017, pp. 36–41.

[5] L. M. Vaquero and L. Rodero-Merino, “Finding your way in the fog: Towards a comprehensive definition of fog computing,” SIGCOMM Comput. Commun. Rev., vol. 44, no. 5, pp. 27–32, Oct. 2014.

[6] S. Natarajan, R. Krishnan, A. Ghanwani, D. Krishnaswamy, P. Willis, and A. Chaudhary, “An analysis of lightweight virtualization technologies for nfv,” 2017. [Online]. Available: https://tools.ietf.org/html/draft-natarajan-nfvrg-containers-for-nfv-03

[7] R. Petrolo, R. Morabito, V. Loscr`ı, and N. Mitton, “The design of the gateway for the cloud of things,” Annals of Telecommunications, pp. 1–10, 2016.

[8] Ali, Hazrat - Smart house demonstration, https://github.com/iloveyii/iot-lab