Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 111 (2017) 260–267

Procedia Computer Science

www.elsevier.com/locate/procedia

8th International Conference on Advances in Information Technology, IAIT2016, 19-22 December 2016, Macau, China

# Container based testbed for gate security using open API mashup

Dongwoo Kwon, Hyeonwoo Kim, Donghyeok An, Hongtaek Ju*

*Department of Computer Engineering, Keimyung University, Daegu, Republic of Korea*

**Abstract**

In this paper, we propose a container based testbed for the gate security system for a smart campus. The security system incorporated two-factor authentication, ID card based identification and face recognition, and authorization which were implemented using an open application programming interface (API) mashup. We analyzed requirements to provide an efficient and flexible testbed to developers and testers working simultaneously. Then, we designed and constructed a container based structure to satisfy the requirements including lightweight virtualization technology and dynamic private domain name management. Finally, we present the automation script to build the testbed and test the security system.


Peer-review under responsibility of the organizing committee of the 8th International Conference on Advances in Information Technology

*Keywords:* Container; gate security system; open API mashup; smart campus; testbed; virtualization.

## 1. Introduction

With the advance of virtualization technology, a number of information and communication technology (ICT) services have been developed based on server virtualization solutions, such as VMware vSphere[1], Citrix XenServer[2], Microsoft Hyper-V[3], and Docker[4]. Virtualization technologies provide scalable hardware resource management to increase hardware resource utilization, leading to cost reduction, efficient service operation, and productivity improvement[5].

We have been developing a gate security system that combines ID card and face recognition based authentication to strengthen security for a smart campus. The essential functions of a gate security system are user authentication

* Corresponding author. Tel.: +82-53-580-5234.
*E-mail address:* juht@kmu.ac.kr

1877-0509 
Peer-review under responsibility of the organizing committee of the 8th International Conference on Advances in Information Technology
10.1016/j.procs.2017.06.062

and authorization, and various open application programming interfaces (APIs) have been employed, such as Face++[6], Lambda Labs[7], Betaface[8], and Kairos[9] for face recognition and biometrics based user authentication; and Stormpath[10] for user management and authorization. This open API mashup enables us to meet the requirements of high service quality for specific functions with low cost, while decreasing research and development spending.

To develop and test the gate security system, we also built a testbed using Linux containers[11,12] as the virtualization technology. The gate security system consisted of eight modules, including face recognition, tag detection, and role based access control (RBAC) modules. Each module is independent of the others to support scaling, and data exchange between the modules is conducted via Java message service (JMS). Docker was used to build the testbed, because the containers, which share the kernel and part of the operating system (OS) without a hypervisor, have higher performance and scalability[13,14] than virtual machines[12]. It also allows simultaneous execution and testing of multiple module sets on a physical machine for a number of developers and testers, and frequent start-up and shutdown operations of the module containers.

For efficient and flexible development and testing of the gate security system, the testbed should provide dynamic domain name management, interoperation of the software configuration management (SCM) system and testbed, integrated log analysis, public IP address assignment for the module containers, etc. This paper analyzes the requirements of the testbed in detail for a gate security system development using open API mashup. We design the container based testbed structure to meet the indicated requirements, and describe the workflow between the modules in the testbed and external services. We discuss and implement the *gss-test* shell script to automatically build the testbed using latest source codes.

The rest of this paper is organized as follows. In Section 2, the gate security system using open API mashup is introduced and discussed. Section 3 analyzes the requirements to build an efficient and flexible testbed. The container based testbed is designed based on the requirements, and the shell script is introduced. Section 4 reviews related work regarding building testbeds, and we conclude the paper and discuss future work in Section 5.

## 2. Gate security system using an open API mashup

A gate security system provides physical security and its essential functions are authentication and authorization. We adopted two-factor authentication, ID card based identification and face recognition, to strengthen security. ID card based authentication has very high accuracy and low complexity, but identity theft risk by acquisition and replication. In contrast, face recognition, i.e., biometrics based authentication, has relatively less opportunity for identity theft, but lower accuracy and very high complexity. Combining these authentication methods, the proposed gate security system reduces identity theft risk and retains high authentication accuracy.

Open API services were employed to implement the face recognition module, including Face++, Lambda Labs, Betaface, and Kairos. An IP camera deployed at the gate detects visitor's faces, and captured images are stored in a file transfer protocol (FTP) server. The images are forwarded to the face recognition service platforms by an internet of things (IoT) adapter module. Face recognition APIs are then applied to images after combining using a simple threshold based ensemble method. The open API mashup for face recognition enables us to use high performance algorithms with low cost.

For ID card based identification, tag detection and tag reader modules on an Arduino board[15] were implemented using the Open Connectivity Foundation (OCF) IoTivity framework[16] for radio frequency identification (RFID) tags. The RBAC module for user authorization was implemented with two-factor authentication utilizing Stormpath, an open API service for user management. The results of the permission by the RBAC module were notified to registered visitors by a short message service (SMS) notification module implemented using an SMS API.

A web interface was provided to register visitors and manage authorization, which included campus management and usage statistics. The web server was separated from the gate security system. The gate security system provides representational state transfer (REST) APIs[17], and the web interface requests data from the module using REST APIs.

To support scaling and load balancing, modules in the gate security system were independent and exchange data via the message router module. The message router module was implemented with Apache ActiveMQ[18] as JMS Queue, and messages are exchanged using pre-defined queues. Message flow between the modules is shown in Fig. 1.