RailCloud: A Reliable PaaS Cloud for Railway Applications

Bijun Li, Rüdiger Kapitza
TU Braunschweig
{bli,rrkapitz}@ibr.cs.tu-bs.de

The railway network is one of the most critical infrastructures of a country. In Germany, in addition to Deutsche Bahn (DB), many small and medium-sized transportation companies (SMTCs) offer railway services for passenger as well as freight transport. However, in contrast to DB, SMTCs are often lack of resources to invest in modern automated control and safety guarantee systems. When it comes to regional railway networks with relatively low traffic, the lack of economics of scale leads to a disadvantage of competition and increased safety risks (e.g. single-point-of-failure) for the SMTCs.

RailCloud is a reliable and highly available *Platform as a Service (PaaS)* cloud, delivering a computing platform, including operating system, programming-language runtime, database and web server etc., without requiring any resource investments or maintenance tasks of the underlying infrastructure. Besides the cost-effective advantage, RailCloud also provides strong safety guarantee to the SMTCs. A systematic approach that combines advanced software and cutting-edge hardware technologies is developed to deliver reliable cloud services, addressing the security, availability and fault tolerance issues of the deployed railway applications. RailCloud is built upon container-based PaaS clouds, where applications are deployed in containers with essential software runtime environment, and related containers are coordinated to provide services to the clients. The architecture of RailCloud is demonstrated in Fig. 1.

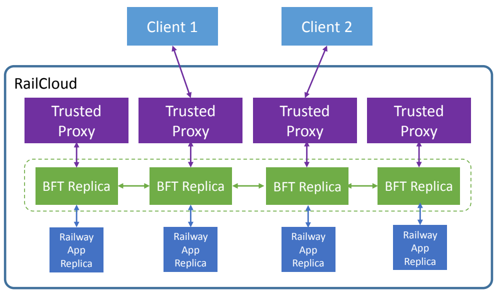

Fig. 1. RailCloud architecture.

RailCloud extends the PaaS clouds by adding a *BFT service* layer and a *trusted proxy* layer to each application (see Fig. 1). The BFT service layer consists of multiple BFT protocol instances (BFT replicas), where each one is connected to a replicated application instance (application replica) and coordinated to reach consensus on the sequence of incoming requests. The trusted proxy layer includes individual proxies to provide secure connections to the application clients as well as trustworthy services to the BFT replicas.

RailCloud is implemented with an open-source cloud OpenShift Origin v3[1] and uses the existing facilities as much as possible. High availability and reliability is achieved by generating replicated application instances and using state machine replication-based protocols to coordinate them. Byzantine Fault Tolerance (BFT) protocol[2], which tolerates Byzantine failures in distributed systems, has been integrated into RailCloud as a built-in service for customers to build the BFT service layer. Replicas of customer applications are created and automatically connected to each BFT replica to execute the ordered requests and eventually deliver fault-tolerant services.

Additionally, RailCloud aims to simplify the deployment of legacy railway applications. It offloads the BFT client library and relocates it in each trusted proxy inside RailCloud to make the replicated systems *transparent* to the application's clients. Therefore most of the web-based applications can be deployed in RailCloud without modifications, making it very friendly to the low-bandwidth clients and secure as well in terms of hiding server side details. In RailCloud, a hardware-based secure component from Intel, called Software Guard Extensions (SGX)[3] is used for building the trusted proxies. Each proxy has the following functions: 1) It provides common secure socket connections that cannot be forged by malicious replicas, for the clients to transfer each single request or reply message. 2) It protects the integrity of the BFT client library functions, such as request distributing and reply voting, from being manipulated by malicious replicas. 3) Moreover, it maintains a fast-read cache for the write-operation requests, and performs fast reads if the cached data for a read request exist. All the write operations are forced by RailCloud to keep the cached data up-to-date, to avoid continuous stale reads from malicious replicas. This way, RailCloud is able to provide *linearizability* semantics to the replicated services and reduce cost of replicated system, which makes it attractive to the SMTCs to deploy their critical applications.

This work is supported by Siemens international Rail Automation Graduate School (iRAGS).

[1]Openshift Origin v3. https://github.com/openshift/origin

[2]Castro, M., Liskov, B.: Practical byzantine fault tolerance. In: Proc. of the 3rd USENIX Symp. on Operating Systems Design and Implementation (OSDI 99). pp. 173186 (1999)

[3]Intel Software Guard Extentions. https://software.intel.com/en-us/sgx