Week 11

This week I learn about Logically Centralized Architectures called ONOS.

ONOS provides two prototypes of a software-defined multicontroller model, which differ in many aspects.

The first prototype has three characteristics: the global network view, the scalability, and the fault tolerance. This prototype keeps a global network view by gathering switch, port, and link information.

The network view has three components: Titan (a graph database), Cassandra (a key-value store), and Blueprints(a graph API to expose network state to the application layer).

ONOS can add supplementary instances to distribute the workload on the control plane when it is scaling out.

ONOS can reassign a task to another instance to prevent failures.

The results of the evaluation study of ONOS prototype 1 showed that ONOS can control hundreds of switches and hosts. Moreover, ONOS can add dynamically and efficiently switches and instances and deal instantly with network failures.

ONOS presents a decent level of consistency and integrity, because it uses Titan that maintains the graphs’ structural integrity and Casandra, which has a high level of consistency.

The first problem of prototype 1 is excessive data store operations. In other words, the task of mapping data, from the Titan graph to Cassandra, results in a significant number of data store operations, which slows the network. The second problem is the lack of notification and messaging system, which is essential for the proper communication between the controllers.

The second prototype focuses on improving the performance of the first prototype, while keeping the global network view consistent. Since the main problem of the first prototype was an excessive data store operation, in prototype 2, the authors will try to solve this issue, following two complementary approaches. The first one concerns making remote operations as fast as possible, while the second approach focuses on reducing the number of remote operations.

Following the first approach, they implemented the Titan/Cassandra system with a Blueprints graph implementation on the top of a data store called: RAMCloud, which has a low latency of order of 15–30 μs.

Following the second approach, they created a cache topology system. This way some of the remote data store operations are stored in the memory cache. Likewise, they can reduce the number of data storage operations globally in the system.

To remove the problem of notifications intercontrollers, the authors adopted a communication system based on Hazelcast. These communications will go through some channels installed at the top of all instances of the control plane.