Deduplicating Cloud Functions - Demo 1

Mentor: **Shripad Nadgowda** (nadgowda@us.ibm.com)

Beliz Kaleli
bkaleli@bu.edu

Vikash Sahu
vksahu@bu.edu

Paritosh Shirodkar
paritosh@bu.edu

Asutosh Patra
asupat12@bu.edu

BOSTON UNIVERSITY

Project Overview

Aim:

- Develop **function de-duplication** framework for serverless platform

Our Clients:

- Cloud users who use serverless computing
- Cloud vendors

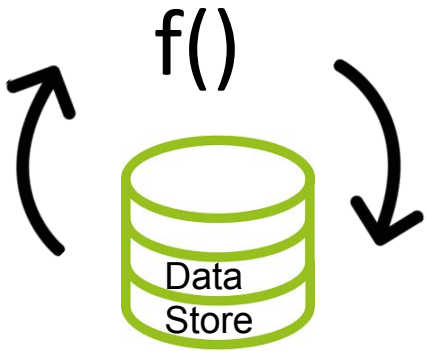
What is Serverless?

- A way to build and run applications and services without having to manage infrastructure
- Application still runs on servers
- Cloud Provider does: server management for applications and databases, provisioning, scaling

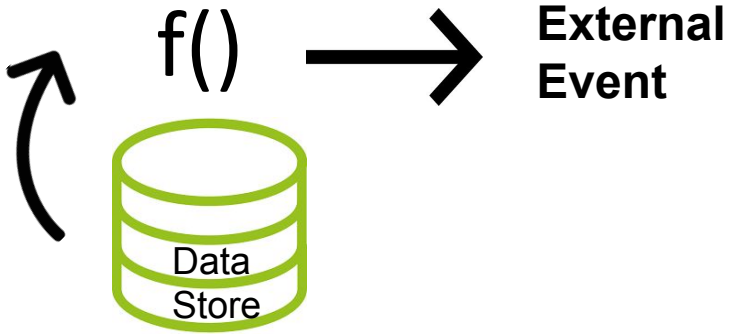
Serverless Functions

- Stateless and idempotent
- Types of functions:

1. Storage Closed Loop



2. External Stimuli



How does Serverless Work?

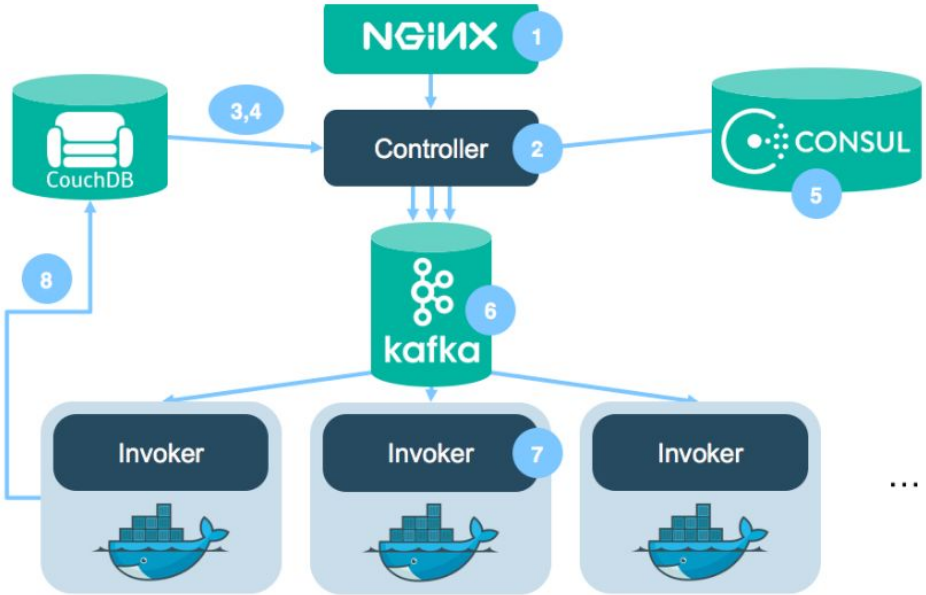


Fig 1. OpenWhisk architecture, Retrieved from:

https://medium.com/openwhisk/uncovering-the-magic-how-serverless-platforms-really-work-3cb127b05f71

Steps:

- SSL Terminator forwards HTTP Req.(user command)
- Check if user exists in OpenWhisk and their privileges
- Consul finds available invokers
- Controller chooses one invoker
- Controller publishes message to Kafka(action+parameters)
- ActivationId is sent to user
- A Docker container spawned, code injected, executed, result obtained, container destroyed
- Result is stored in DB

Demo

Real world Use Case

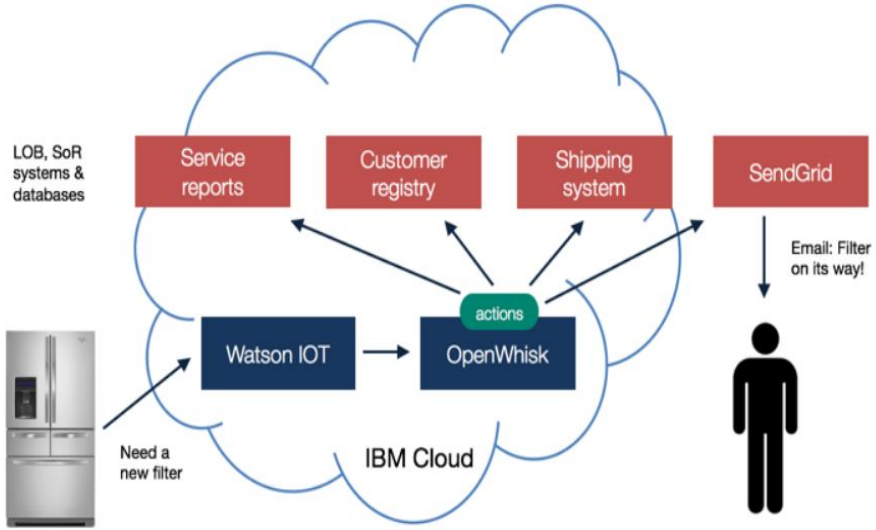


Fig 2. IoT Openfridge Use-case Example, Retrieved from: https://github.com/IBM/ibm-cloud-functions-serverless-iot-openfridge

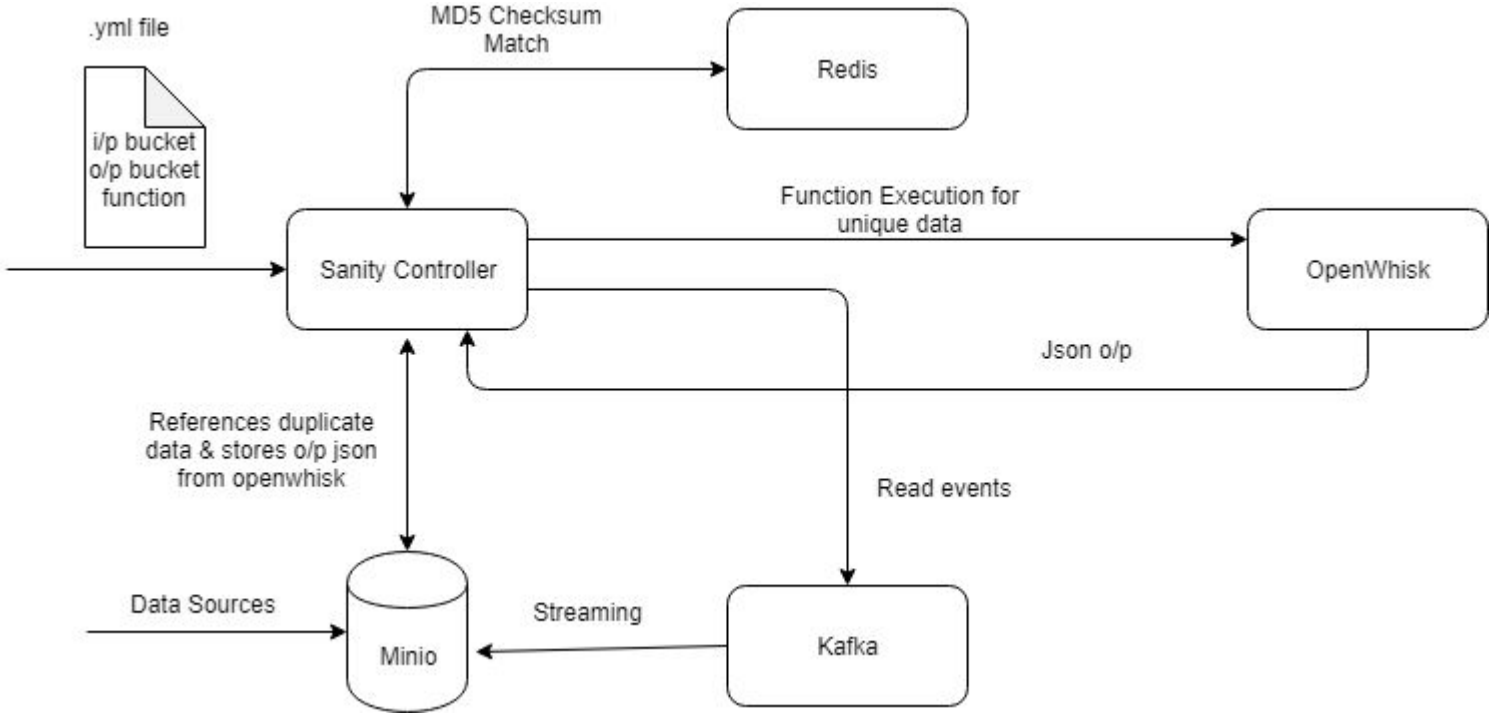
- Trigger:Collect sensor data on a daily basis
- Actions:Run service diagnosis event daily
- Rules: If an anomaly occurs, run an event which determines the need of a new filter event

Motivation

Most cloud platforms today follow the ‘pay only for what you use’ model i.e. you are charged based on the number of requests for your functions and the duration, the time it takes for your code to execute.

But what if we could reduce the number of requests for functions?

Proposed Architecture



Exploration of Sanity Controller

- The survey for Sanity system was evaluated with its performance and resource overhead for one real-world use case of continuous container vulnerability analysis.
- During this research, with modest overhead of 2GB memory it can help improve the latency of function for duplicate data by about 200x.
- Reference: http://niltonbila.com/pub/Nadgowda-WoSC17.pdf

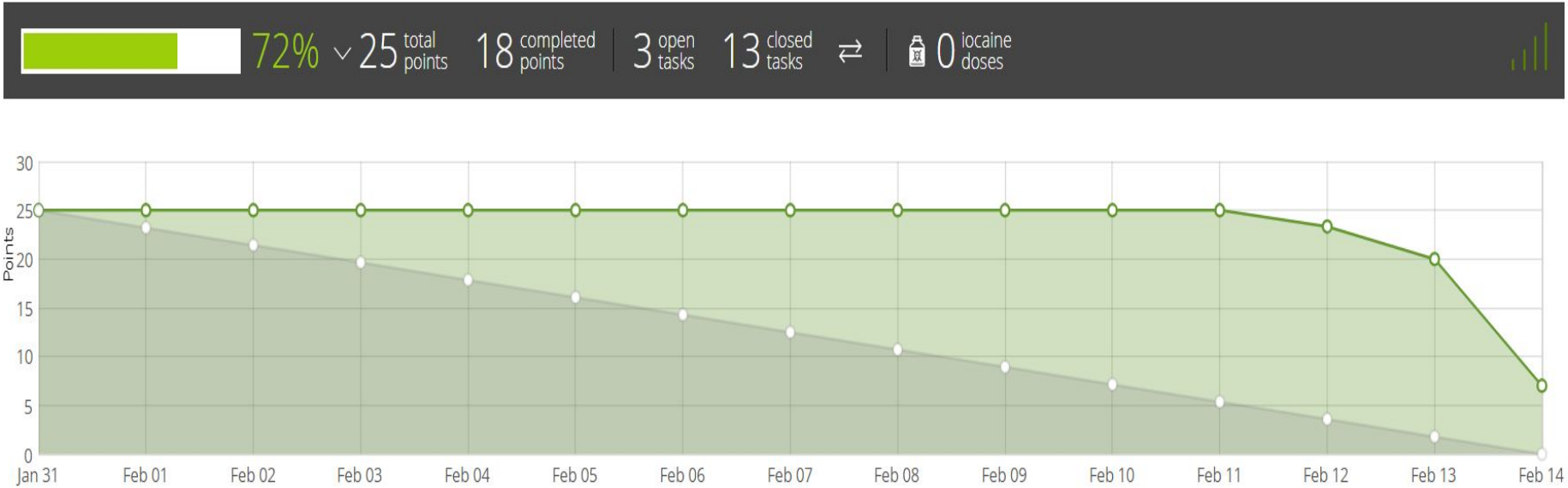
Scope

- Our intention is not to deduplicate data but to deduplicate invocation of cloud functions
- We are going to make a POC on top of Openwhisk platform

Current Challenges

- To containerize each application and run serverless computation to identify duplicate functions
- Integrate each components in Sanity architecture.

Burndown Chart



Next Sprint

- Integrate a Proof of Concept (POC) for the Sanity system for deduplication of cloud functions
- Initiate a full trigger of Openwhisk with reference to serverless architecture
- Better estimation of our tasks and a better burndown pace

References

[1](n.d.). Retrieved from https://tree.taiga.io/project/bowenislandsong-deduplicating-cloud-functions/taskboard/sprint-1-13886Ibm. (2018, December 12).

[2]IBM/ibm-cloud-functions-serverless-iot-openfridge. Retrieved from https://github.com/IBM/ibm-cloud-functions-serverless-iot-openfridge

[3]Sanity: The Less Server Architecture for Cloud functions. (n.d.). Retrieved from https://www.serverlesscomputing.org/wosc2/presentations/p4-Sanity-WoSC_v0.pdf

[4]Thömmes, M., & Thömmes, M. (2016, October 11). Uncovering the magic: How serverless platforms really work! Retrieved from https://medium.com/openwhisk/uncovering-the-magic-how-serverless-platforms-really-work-3cb127b05f71

Thank You