A Design of Failure Injection Testing considering Edge Computing Environment

Kenta Hayashi†, Kaori Maeda†, Tohru Kondo‡

† Graduate School of Information Sciences, Hiroshima City University, Japan.
‡ Information Media Center, Hiroshima University, Japan.

Background

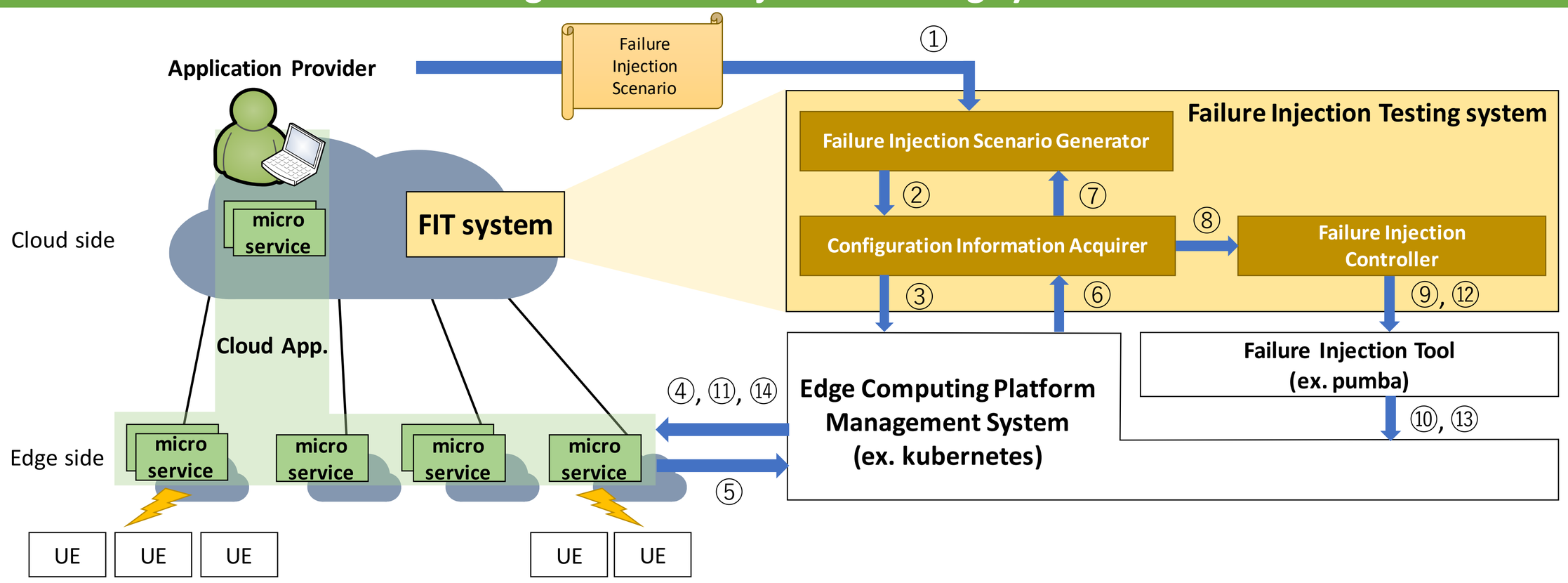
- The current cloud services composed by many distributed micro services[1]
- High complexity of collaboration of micro services
- Requirement for availability and capability without service disruption
- Emerging of Chaos Engineering[2] to improve for resilience of complex distributed systems
- Demand of edge computing for IoT
- Long latency or unstable connections in an edge computing environment have many negative effects on IoT applications
- Edge computing environments including IoT devices have different failure occurrence rates depending on places

The goal of this research

Implementation of a Failure Injection Testing(FIT) system for edge computing environment

- Design of a failure injection scenario which is friendly to application providers
- Implementation of the FIT system based on an arbitrary scenario such as power failure into 50% of edge servers

Design of failure injection testing system



UE (User Equipment) : Edge devices such as sensors

Fig. FIT system configuration

Failure injection flow:

Step 1 (①-⑦) : Acquisition of application configuration information

Step 2 (⑧) : Failure injection preparation

- Creation of failure injection scenario based on configuration information in Step 1

Step 3 (⑨- ⑪) : Failure injection testing based on the scenario which defined in Step 2

Step 4 (⑫ -⑭) : Restoration to the original state

Failure injection scenario

- Indicates structured scenarios for the edge computing environment including the following items
  - Failure injection range (ex. edge side server, access network)
  - Type of failure (ex. packet loss, jitter)
  - Probability of failure occurrence (ex. 50%)
  - Failure injection period (ex. 10 minutes)

Failure injection function

- Micro service operation
  - Micro service stop
    - Slow stop
    - Immediate stop
    - Pause
  - Micro service removal
- Network emulation
  - delay
  - packet loss
  - duplicate
  - corrupt

Future prospects

- Implement a FIT system using pumba[3] for failure injection and kubernetes[4] for edge computing environment
  - Demonstration of the effectiveness of the developed FIT

Reference

[1] Jim Gray, “Why do Computers Stop and What Can Be Done About It?,” Tandem Computers Technical Report 85.7, PN87614, June 1985.
[2] Ali Basiri, Niosha Behnam, Ruud de Rooij, Lorin Hochstein, Luke Kosewski, Justin Reynolds, Casey Rosenthal, “Chaos Engineering,” IEEE Software, vol.33, no. 3, pp. 3541, May 2016.
[3] “pumba,” https://github.com/alexei-led/pumba, (accessed 08/24/2018).
[4] “Production-Grade Container Orchestration, ” https://kubernetes.io/, (accessed 08/24/2018).

This research and development work was supported by the JSPS KAKENHI Grant Number 15K00130, 16H02808, 18K11266 and MIC/SCOPE #162108102.