Kafka-Python SMS Stream Controller

Project Requirements:

1. SMS gateway max throughput is 2 requests per second
2. Multiple SMS requests per single alert
3. Multiple tenants send SMS requests to single SMS gateway
4. To build Kafka message queue system
5. Concentrate the SMS requests from multiple tenants and send out the requests to SMS gateway with flow control
6. To optimize the platform SMS mechanism
7. To be a reference for integration between thingsboard and other microservices

Prerequisites Evaluation:

Check Kafka can buffer high concurrent messages stream in correct order.

Check if can leverage kafka-python module to pull messages from kafka message queue in correct order.

Procedure & Test Plan:

1. Implement Zookeeper and Kafka broker in UAT platform for evaluation to consolidate an environment requirement and configuration 🗸
2. Implement Kafka producer in UAT thingsboard to publish SMS request message 🗸
3. Develop a Python program of Kafka consumer to pull the SMS request message, reassemble the request and send to UAT thingsboard for logging 🗸
4. Test and monitor if the producer and consumer behaviors are aligned and stable 🗸
5. Create a new tenant to publish SMS request message together with original tenant, test if multi-tenant SMS behavior 🗸
6. Add queue status monitoring system, if there is a significant growing trend of the queue, send the alert.
7. Add input validation for the incoming data stream, avoiding buffer overflow.
8. Add the mechanism to delete the queue message after committing, to saving the server storage.
9. Modulize the kafka consumer program for the further reuse in addition to SMS (eg. Analysis, outside rule engine).
10. Deploy the whole set of this SMS stream controller to Production platform
11. Use dummy device to trigger SMS request to Kafka, test and monitor if the producer and consumer behaviors are aligned and stable
12. Optimize the consumer program with input validation and robustness
13. Apply new SMS gateway for Production platform for test and demo purpose
14. Modify the consumer program to send SMS request to new SMS gateway, test and monitor if the triggered alarm and received SMS message are aligned and stable
15. Create a new tenant for demo account
16. Publish demo account SMS request message together with original tenant, test if multi-tenant SMS behavior
17. Evaluate and finalize a device & data migration plan to migrate the devices and asset from original tenant to new tenant without data lose
18. Create new tenants for projects in Production platform and implement the devices & assets migration
19. Test and monitor if the SMS stream controller work normally and no data lose of the migration

Development log:

30 Sept [Nan] Installed & configured Kafka in UAT server for evaluation (1 man day)

9 Oct [Nan] developed a draft python program to consume Kafka message (3 man day)

22 Oct [Nan] optimize consumer program and integration with UAT Thingsboard. Generate simulated device message and keep monitoring the program and system stability (2 day)

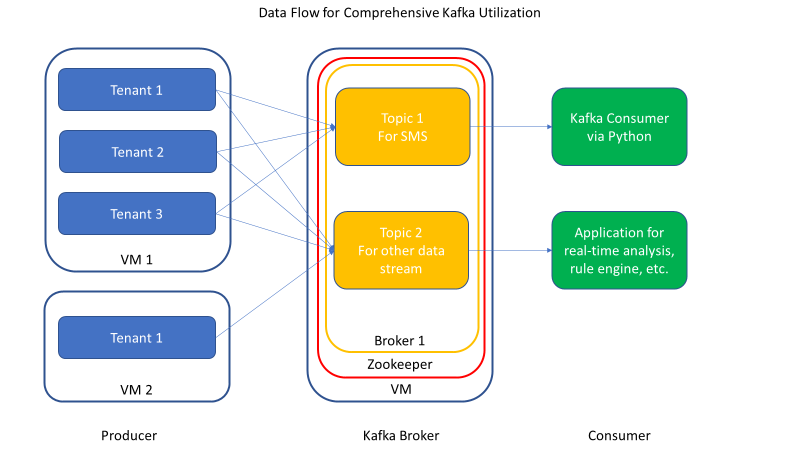
5 Nov [Nan] modify SMS request flow from 2 request per 2 second to 1 request per second for easier flow control and logging (0.5 day)

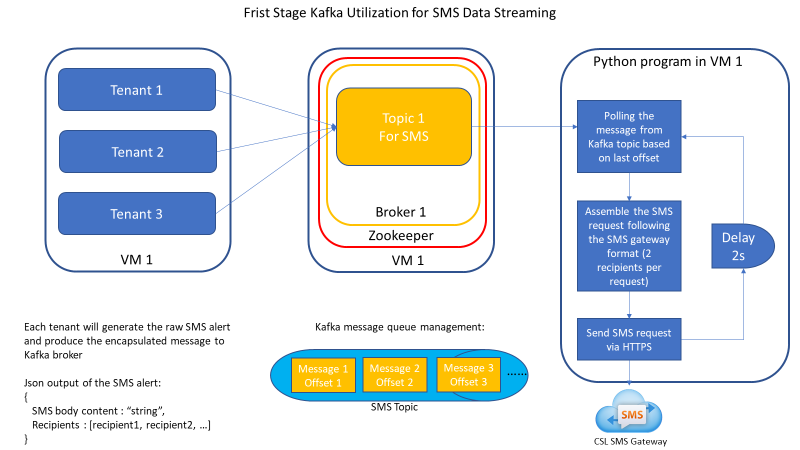
6 Nov [Nan] preparing test plan & deployment procedure (0.5 day)

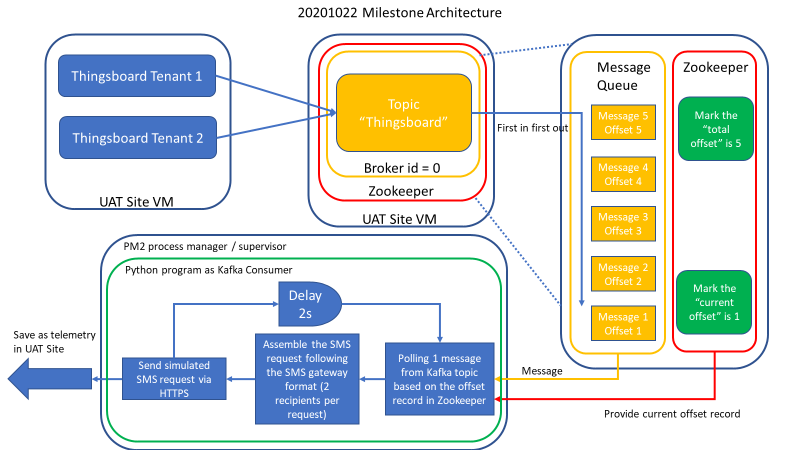
9 Nov [Nan] finalized test plan & deployment procedure, prepared development plan (1 day)

Development Deliverables & Test Result:

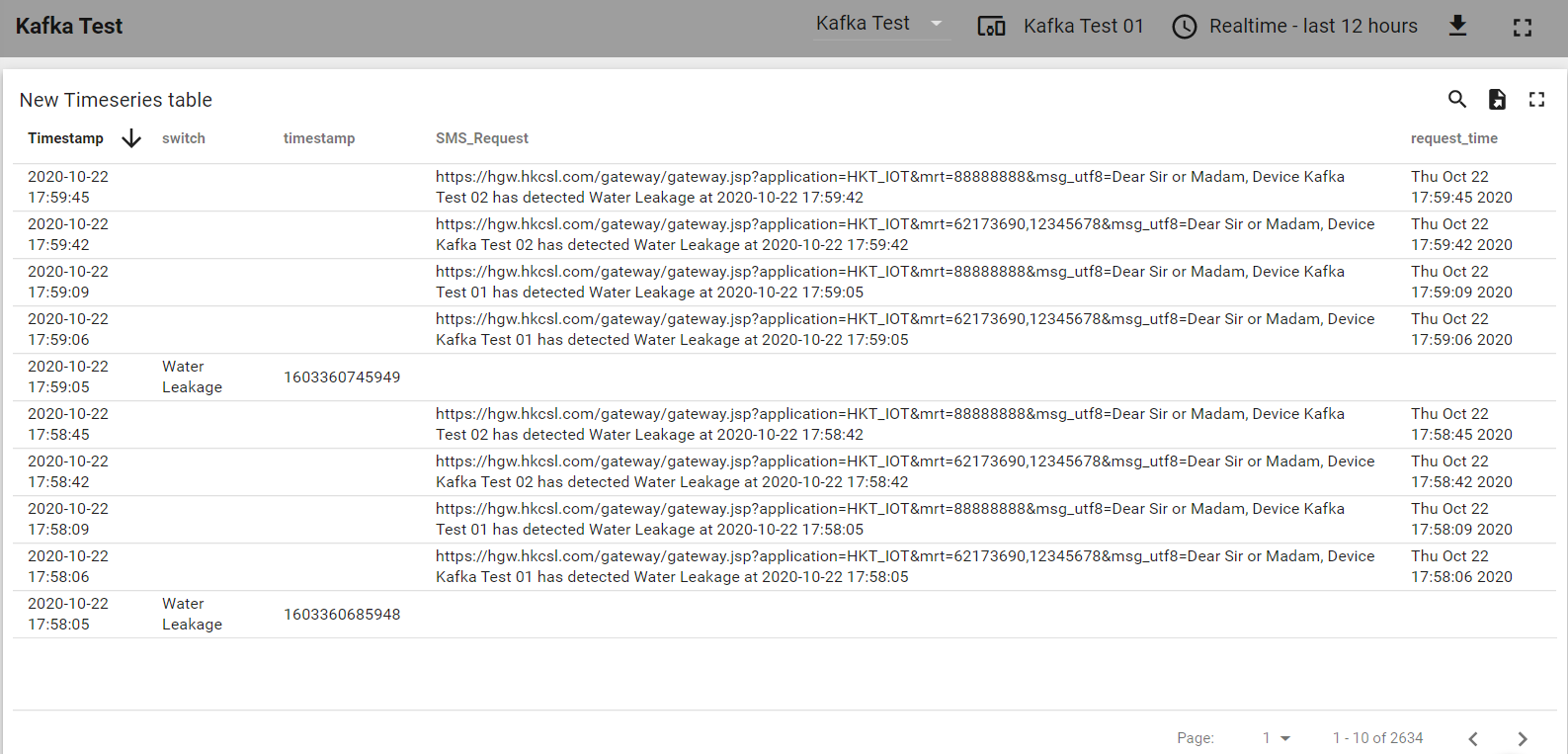
***2020 Oct 23 Milestone:***







Architecture in UAT



The producer and consumer behaviors are aligned and stable in weeks long period