IBM ROKS EU Development Environment – Smoke Testing Plan

**Introduction**

Recently SRE team started working with a IBM ROKS environment in EU Frankfurt. Need is to provide an alternative to Automation Solution teams for their current Boomerang ICP based Test Environment in RTP4, which is set for take down by end of September 2021. To simulate the existing RTP4 environment, SRE is trying to fetch almost all features over ROKS in its current working plan, where few are already done, while rest all in progress. Here is some Smoke Testing plan for ongoing environment setup plans.

We will do smoke testing for following parts of the newly prepared ROKS environment

1. **ROKS Cluster**

Installing a small test app to check its availability from cluster using cluster load balancer IP and domain name.

1. **ROKS Registry**

Checking whether images are being pushed into registry and scanned properly after docker image build stage pipeline job. This can be done after CI/CD pipeline is ready (4).

1. **Essentials Setup**

Testing Essentials by creating a fresh team, adding members, including a sample service, checking metering and all.

1. **CI/CD Setup**

Deploying a sample application in CI/CD pipeline to check it’s pass through across different jobs in pipeline, e.g., fetch from code repository, SAST via SonarQube, scanning through Xray, pushing image in Artifactory, Deploying in ROKS Cluster, etc. An end to end check with manual effort.

1. **3rd Party Services (MySQL/MongoDB/RabbitMQ/Redis/PostgreSQL/Camunda)**

Testing each 3rd party services by its own proprietary testing process

1. MySQL: create a DB, a table inside DB, inserting data in table, creating user with credentials, allow access to DB, check access with credentials, apart from normal Admin access, which SRE should hold for troubleshooting. Finally setting up Dashboard Grafana for MySQL DB instance and check if the performance is reflected over there.
2. MongoDB: create a DB, a table inside DB, inserting data in table, creating user with credentials, allow access to DB, check access with credentials, apart from normal Admin access, which SRE should hold for troubleshooting. Finally setting up Dashboard Grafana for MongoDB instance and check if the performance is reflected over there.
3. RabbitMQ: Create a test queue, exchange, and routing key to send a message in system and check its progress from end to end. Finally setting up Dashboard Grafana for RabbitMQ instance and check if the performance is reflected over there.
4. Redis: Set a dummy message in Redis and retrieve the same through a sample application in different mutually disconnected services, to check if Redis really stores it in cache. Finally setting up Dashboard Grafana for Redis instance and check if the performance is reflected over there.
5. PostgreSQL: test with a sample DB, sample table, sample data, sample user with credentials and connectivity with the same, apart from Admin access (SRE should hold it). Finally setting up Dashboard Grafana for PostgreSQL instance and check if the performance is reflected over there.
6. Camunda: Need solution team to create a test flow and test for all.
7. **EFK Setup**

Create a sample application, team, members, to check if it is reflecting its logs into EFK stack’s Kibana dashboard and viewable from Admin and Team via granted restricted access

1. **Vyatta Installation**

Routing ROKS cluster’s public VLAN via IBM VRA (Vyatta Router), using “Route Through” option to test whether all applications are accessible through Vyatta public IP address.

1. **Prometheus / AlertManager / Grafana Setup**

Setup dashboard for Grafana to check whether some common features like Cluster Health, POD Health, PVC Health, Health for 3rd Party Services and Golden Signal for a Test Solution. Check all the dashboards from Admin. Setup a sample team and add few members from SRE to allow check restricted access for them, like watching some common dashboards, dashboards for solutions belonged to that team, etc., with augmented power. Testing Email notification against set rules in solution specific way.

1. **Service Mesh**

Creating a sample application to test them in Jaeger, Kiali and Mesh Console, after putting a load on the sample application to let it dispose some metrics of significance. Jaeger installation is priority, which is dependent on Elasticsearch.

Progress of each of the above tests and their current status will be published as part of our daily task status, soon.