Overview of the Service Assurance architecture

Gerardo García (Telefónica, OSM TSC Chair)

OSM Training Seminar - SLICES

13/02/2024
Release FOURTEEN comes with the new Service Assurance architecture enabled by default

- MON and POL functionality has been transferred to the new architecture
  - Metric acquisition (VM resource consumption)
  - Closed-loop for auto-healing
  - Closed-loop for auto-scaling
  - VNF alarms

- Airflow, Prometheus AlertManager and PushGateway are deployed
  - Airflow provide a mechanism to run scheduled workflows, as well as workflows on demand from an alert

- Webhook Translator is a new component that has been added to translate webhooks
The new SA architecture is installed by default

/install_osm.sh

$ helm -n osm ls
NAME            NAMESPACE       REVISION        UPDATED                                 STATUS          CHART             APP VERSION
airflow         osm             1               2023-06-07 15:00:48.61309036 +0000 UTC deployed        airflow-1.9.0     2.5.3
alertmanager    osm             1               2023-06-07 15:10:23.448897581 +0000 UTC deployed        alertmanager-0.22.0 v0.24.0
osm             osm             1               2023-06-07 15:08:43.421836769 +0000 UTC deployed        osm-0.0.1         14
pushgateway     osm             1               2023-06-07 15:10:19.507304535 +0000 UTC deployed        prometheus-pushgateway-1.18.2 1.4.2

$ kubectl -n osm get pods
NAME                                                  READY   STATUS    RESTARTS        AGE
airflow-postgresql-0                                  1/1     Running   2 (2d20h ago)   5d22h
airflow-redis-0                                      1/1     Running   1 (2d20h ago)   5d22h
airflow-scheduler-5f7dbd4c4f5-54x9c                  2/2     Running   4 (2d20h ago)   5d22h
airflow-statsd-d8cs8f886c-vt7xp                       1/1     Running   4 (2d20h ago)   5d22h
airflow-triggerer-66680b865c-n6snh                    2/2     Running   3 (2d20h ago)   5d22h
airflow-webserver-5fb957dcf7-bcgoz                    1/1     Running   1 (2d20h ago)   5d22h
airflow-worker-0                                     2/2     Running   2 (2d20h ago)   5d22h
alertmanager-0                                       1/1     Running   6 (2d20h ago)   5d22h
grafana-69c9c55dfb-jtfw1                              2/2     Running   2 (2d20h ago)   5d22h
kafka-0                                               1/1     Running   1 (2d20h ago)   5d22h
keystone-7dbf4b7796-rqwe4                              1/1     Running   1 (2d20h ago)   5d22h
k8s-lcm-6d97b88675-4m77j                               1/1     Running   2 (2d20h ago)   5d22h
modeloperator-7d88b06fc79-ww49m                       1/1     Running   1 (2d20h ago)   5d22h
mongodb-cc0965d54-driver                               1/1     Running   1 (2d20h ago)   5d22h
mongodb-k8s-0                                        1/1     Running   3 (2d20h ago)   5d22h
mongodb-k8s-operator-0                                1/1     Running   1 (2d20h ago)   3d11h
mysql-0                                               1/1     Running   1 (2d20h ago)   5d12h
mysql-0-nbi-6b4f66fd9-jtbf5                            1/1     Running   5 (2d20h ago)   5d22h
pkgy-78d9b6d6bc-xb6f6                                  1/1     Running   3 (2d19h ago)   5d22h
prometheus-0                                          2/2     Running   4 (2d20h ago)   5d22h
pushgateway-prometheus-pushgateway-6f9dc6cb4d-4sp4x     1/1     Running   1 (2d20h ago)   5d22h
ro-86cf79d555-z61s1                                   1/1     Running   5 (2d20h ago)   5d22h
webhook-translator-57b75fc797-j9s7w                    1/1     Running   1 (2d20h ago)   5d22h
zookeeper-0                                           1/1     Running   1 (2d20h ago)   5d22h

© ETSI
Building blocks
Building blocks of the new SA architecture

Apache Airflow + Prometheus Stack
Building blocks of the new SA architecture

Apache Airflow

- Webserver
- Scheduler
- Workers
- DAGs

Data Sources

Output

DAG Example

© ETSI
Building blocks of the new SA architecture
Airflow DAGs

- DAG (Directed Acyclic Graph):
  - Collection of tasks
    - A lot of flexibility to create dependencies between tasks
  - Defined in Python
  - DAGs can be dynamically created, for instance:
    - One per VIM
    - One per NS
  - Tasks can be dynamically created inside a DAG, for instance:
    - One per VM
  - Designed to scale
    - Airflow workers run tasks in parallel
  - Scheduled independently
Building blocks of the new SA architecture

Prometheus Stack

- Prometheus
- Push Gateway
- Prometheus
- Push Alerts
- Alert Manager
- Pull Metrics
- Pull Metrics
- Pull Metrics
- Pull Metrics
- Pull Metrics
- Push Metrics
- Push Metrics
- Push Metrics

- VIM Exporters
- VNF Exporters
- Grafana
Building blocks
Webhook translator

Alert Manager
- vdu_down
- scalein_vdu
- scaleout_vdu
- vdu_alarm

xNFs
- vdu_down
- scalein_vdu
- scaleout_vdu
- vdu_alarm

VIMs
- vdu_down
- scalein_vdu
- scaleout_vdu
- vdu_alarm

External systems (e.g. OSS)
- vdu_down
- scalein_vdu
- scaleout_vdu
- vdu_alarm

Webhook Translator

POST api/v1/dags/<endpoint>/dagRuns

- endpoint = vdu_down
- endpoint = scalein_vdu
- endpoint = scaleout_vdu
- endpoint = vdu_alarm
- endpoint = endpoint5
- endpoint = endpoint4
- endpoint = endpoint5
- endpoint = endpoint6
- endpoint = endpoint7

Airflow
- DAG vdu_down
- DAG scalein_vdu
- DAG scaleout_vdu
- DAG vdu_alarm
- DAG endpoint5
- DAG endpoint6
- DAG endpoint7

© ETSI
Building blocks
Webhook translator

● Principles:
  ● Lightweight: a very small number of lines of code will do the work.
  ● Stateless. It only translates HTTP requests. No state for those translations
    ● When running as a deployment, native scaling is achieved by means of Kubernetes services
  ● Simple. Based on FastAPI (https://fastapi.tiangolo.com/)
    ● Simple and fast framework for developing an HTTP REST API in Python.
  ● Independent from the source of the alert
    ● No maintenance
Workflows
Workflow for metric acquisition and derivation
Metric acquisition

- **NS topology:**
  - From MongoDB to Prometheus
  - SW used: Airflow DAG + Prometheus PushGateway

- **VM status:**
  - From MongoDB and VIM to Prometheus
  - SW used: Airflow DAG per VIM + Prometheus PushGateway

- **VIM status**
  - From MongoDB and VIM to Prometheus
  - SW used: Airflow DAG per VIM + Prometheus PushGateway

- **VM metrics (resource consumption)**
  - From MongoDB and VIM to Prometheus
  - SW used: Airflow DAG per VIM + Prometheus PushGateway
Screenshot of Airflow DAGs
Metric derivation

- **Extended VM status:**
  - From Prometheus (NS topology, VM status) to Prometheus
  - SW used: Prometheus Recording Rules

- **VNF status:**
  - From Prometheus (Extended VM status) to Prometheus
  - SW used: Prometheus Recording Rules

- **NS status:**
  - From Prometheus (Extended VM status) to Prometheus
  - SW used: Prometheus Recording Rules
Screenshot of the derived metrics in Prometheus
Closed loops with new SA architecture

Prometheus

Prometheus

Alert Manager

Webhook Translator

Kafka

Airflow

MongoDB

Alerting Rules

Push Alerts

Push Alerts

Push Alerts

Send Actions

Pull Data

Push Alerts

© ETSI
Closed loops with new SA architecture

- Prometheus alerts
  - [https://prometheus.io/docs/prometheus/latest/configuration/alerting_rules/](https://prometheus.io/docs/prometheus/latest/configuration/alerting_rules/)
  - Alerts are automatically triggered and stopped by Prometheus depending on their defining rule

- AlertManager
  - [https://prometheus.io/docs/alerting/latest/alertmanager/](https://prometheus.io/docs/alerting/latest/alertmanager/)
  - Provides a mechanism to send alerts to webhooks
  - In addition, it includes mechanisms for silencing, inhibition, aggregation, etc.

- Webhook Translator.
  - Rationale: AlertManager send alerts with a format that cannot be consumed directly by Airflow DAGs
  - It receives HTTP POST messages from AlertManager and forwards it to an Airflow webhook
Closed loops with new SA architecture

- **Airflow**
  - Runs DAGs driven by webhooks
  - Fed with the information in MongoDB
  - Behaves as Policy Manager (POL). It translates the alert to an action:
    - A Kafka message to be consumed by LCM (heal, scale)
    - Or potentially other actions in the future

- **MongoDB**
  - Keeps persistence of closed-loop actions
  - Previously managed by MON, now managed by LCM
Future work
Future work

- Run vulnerability scan on NG-SA
- Remove installation of old SA from the installer
  - Currently it is an option (--old-sa)
- Move MON functionality (grafana-dashboader) somewhere else
- Refactor DAG code to re-use as much as possible
- Independent time period for different VIMs, probably as a VIM parameter passed during VIM registration
Thank You!
**Workflows**

**Metric acquisition (ns_topology)**

- **Input:** Data Source: NS Record
- **Output:** Acquired Metric (ns_topology)

**Metric acquisition (vim_status)**

- **Input:** Data Sources:
  - VIM Record
  - VIMs
- **Output:** Acquired Metric (vim_status)

**Metric acquisition (vm_status)**

- **Input:** Data Sources:
  - VIM Record
  - VIMs
- **Output:** Acquired Metric (vm_status)
Workflows

Metric derivation (vm_status_extended)

Prometheus

Recording Rules

Input. Acquired metrics:
- vm_status
- ns_topology

SW for the process:
- Prometheus recording rule

Output. Derived metric:
- vm_status_extended

Metric derivation (vnf_status)

Prometheus

Recording Rules

Input. Acquired metrics:
- vm_status_extended

SW for the process:
- Prometheus recording rule

Output. Derived metric:
- vnf_status

Metric derivation (ns_status)

Prometheus

Recording Rules

Input. Acquired metrics:
- vm_status_extended

SW for the process:
- Prometheus recording rule

Output. Derived metric:
- ns_status