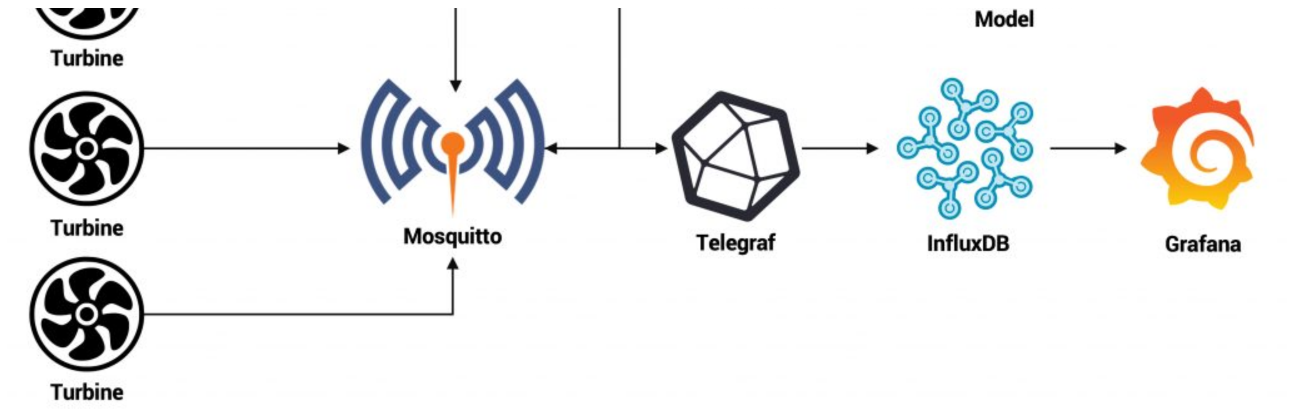
EDGE / IOT / KUBERNETES / MACHINE LEARNING

Tutorial: Configure, Deploy an Edge Application on Cloud Native Edge Infrastructure

5 Oct 2020 1:55pm, by Janakiram MSV

This is the last installment of the 4-part series (Part 1) (Part 2) (Part 3) where we configure and deploy an artificial intelligence workload running on an "edge" Internet of Things (AIoT) workload on cloud native edge infrastructure. The application does predictive maintenance of turbines. For the background and explanation of the use case, refer to the previous part of the tutorial.



Start by cloning the GitHub repository that contains the code, configuration, and Kubernetes deployment manifests. Feel free to explore the source code, configuration, and manifests available in the repository. You can build container images for each of the services to store them in a private registry.

As we start deploying each microservice, I will discuss the design decisions and configuration choices. If you have a K3s cluster configured with Project Calico and Portworx, you can deploy the entire application without building the images.

```
kubectl apply -f deploy/
```

If you want to deploy one artifact at a time, start with the namespace. The workload runs in a dedicated namespace, `aiot`.

```
kubectl apply -f deploy/aiot-ns.yaml
```

Deploying the Mosquitto MQTT Broker

The Mosquitto pod acting as the MQTT broker relies on a persistent volume claim to persist logs and data. This PVC uses dynamic provisioning based on a Portworx storage class which is optimized for databases. The same storage class is used for the InfluxDB time-series database.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
```

```
    repl: "3"
```

Deploy the storage class followed by the Mosquitto service.

```
kubectl apply -f deploy/db-sc.yaml
```

```
kubectl apply -f deploy/mosquitto.yaml
```

Deploying Fan Simulators

The fan simulators are configured as pods with environment variables pointing to the MQTT broker and the topic associated with the telemetry. The environment variable `FAULT` will decide if the simulator publishes anomalous data. The `DEVICE_ID` variable assigns an arbitrary value to the device identifier.

```
      initContainers:
      - name: wait-for-mosquitto
        image: janakiramm/wait
        args: ["--timeout=60", "mosquitto:1883"]
      containers:
      - name: fan-1
        image: janakiramm/fan
        imagePullPolicy: Always
        env:
          - name: MQTT_HOST
            value: "mosquitto"
          - name: MQTT_TOPIC
            value: "fan/messages"
          - name: FAULT
            value: "0"
          - name: DEVICE_ID
            value: "1"
```

The InitContainer will wait for 60 seconds for the Mosquitto broker to become available before timing out. This will ensure that the pod doesn't experience CrashLoopBackOff while waiting for the dependent service.

```
kubectl apply -f deploy/fan-2.yaml
```

Now, we have two simulators publishing telemetry to an MQTT topic.

Deploying AI Inference and Prediction Service

The inference pod downloads the latest version of the model and exposes that as a REST endpoint. An InitContainer checks for the presence of the model in a well-known directory and pulls the compressed TensorFlow model only if it's missing. This approach avoids downloading and copying the model each time the deployment is scaled.

As the inference service scales, the model needs to be available to multiple pods through a shared filesystem. To enable this, we define a different Portworx storage class with the SharedV4 flag to create a globally shared namespace volume that can be used by multiple pods.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: infer-sc
provisioner: pxd.portworx.com
parameters:
  repl: "1"
  sharedv4: "true"
```

```
kubectl apply -f deploy/infer-sc.yaml
```

After creating the storage class, let's create the inference service.

```
kubectl apply -f deploy/infer.yaml
```

With the inference service in place, we can deploy the predictor microservice that acts as the intermediary between the MQTT broker and the AI model.

```
      containers:
      - name: predict
        image: janakiramm/predict
        env:
          - name: MQTT_HOST
            value: "mosquitto"
          - name: MQTT_DEV_TOPIC
            value: "fan/messages"
          - name: MQTT_PREDICT_TOPIC
            value: "fan/anomaly"
          - name: SCORING_URL
            value: "http://infer:5000/score"
```

```
kubectl apply -f deploy/predict.yaml
```

Since the inference microservice is using shared volume, we can easily scale the number of replicas.

```
kubectl scale deploy/infer --replicas=3
```

Deploying Telegraf and InfluxDB

InfluxDB is used as a time-series database to store the telemetry data coming from the fan simulators and the prediction service. It is configured as a stateful set backed by the PVCs created from the same Portworx storage class used by Mosquitto.

```
kubectl apply -f deploy/influxdb.yaml
```

Telegraf connects InfluxDB with Mosquitto through a configuration file which is created as a Kubernetes config map.

```
kubectl apply -f deploy/telegraf.yaml
```

Check the logs of the Telegraf pod to confirm the flow of messages from Mosquitto to InfluxDB.

```
TELEGRAF_POD=$(kubectl get pods -n aiot  -l app=telegraf -o jsonpath='{.items[0].metadata.name}
```

Deploying Grafana and Configuring the Dashboard

A config map associated with Grafana pod configures InfluxDB as the datasource. This bundling helps us quickly import an existing dashboard.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: grafana-datasources
  namespace: aiot
data:
  influxdb.yaml: |-
    {
        "apiVersion": 1,
        "datasources": [
            {
               "access":"proxy",
                "editable": true,
                "name": "influxdb",
                "orgId": 1,
                "type": "influxdb",
                "database": "fan",
                "url": "http://influxdb:8086",
                "version": 1
            }
        ]
    }
```

Deploy Grafana with the below command:

```
kubectl apply -f deploy/grafana.yaml
```

Copy the content of `fan.json` from the `dashboard` directory, paste it into the textbox, and click the load button.

With the configuration in place, you would be able to access the below dashboard:

Configuring Network Policies

Since the K3s cluster is configured with Calico-based CNI, we can secure the network. A sample policy to prevent access to the AI inference is included in the deploy directory.

```
kubectl apply -f deploy/netpol.yaml
```

Janakiram MSV's Webinar series, "Machine Intelligence and Modern Infrastructure (MI2)" offers informative and insightful sessions covering cutting-edge technologies. Sign up for the upcoming MI2 webinar at http://mi2.live.

Feature image by Dmitrii Bardadim from Pixabay.

FEATURE TUTORIAL

analyses.

Email Address

Subscribe

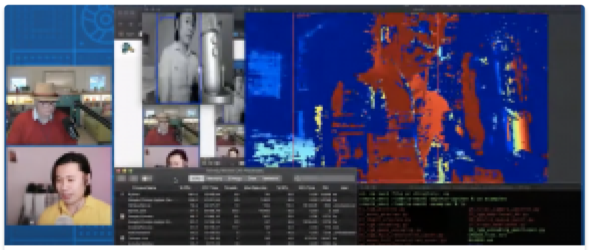
We don't sell or share your email. By continuing, you agree to our Terms of Use and Privacy Policy.

RELATED STORIES

CONTAINERS / KUBERNETES / MACHINE LEARNING

Accelerating Development with Container Run Times, Kubernetes and GPUs

12 Mar 2021 9:16am, by Alex Williams



API MANAGEMENT / DEVELOPMENT / MACHINE LEARNING / SPONSORED

The Developer's Menu for Machine Learning: oneAPI and Your New Hardware

SPONSORED FEED

RED HAT OPENSHIFT

OpenShift Commons Briefing: Deterministic vs Probabilistic Security: Leveraging Everything as Code with Steve Giguere (StackRox)

MARCH 12, 2021

MayaData

WordPress deployment using OpenEBS on DigitalOcean Kubernetes

MARCH 12, 2021

HashiCorp

Boundary Desktop Beta & Boundary 0.1.8 Released

MARCH 12, 2021

CLOUD NATIVE COMPUTING FOUNDATION

Jaeger persistent storage with Elasticsearch, Cassandra & Kafka

MARCH 12, 2021

influxdata

Stargazers and Trailblazers: It's Pi Day!

MARCH 12, 2021

Vates Solutions Open Source

Xen and the RISC-V Hypervisor Extension

MARCH 12, 2021

HAPROXY

Infoblox NEXT LEVEL NETWORKING

HAFNIUM Targeting Exchange Servers with Zero-Day Exploit

MARCH 11, 2021

GitLab

5 Ways to level up your remote engineering skills

MARCH 11, 2021

EQUINIX METAL

Don't Talk About it, Show it

MARCH 11, 2021

aws

How to Use Channel Assembly with AWS Elemental MediaTailor to Launch Virtual Channels from Existing Sources

MARCH 11, 2021

Check Point SOFTWARE TECHNOLOGIES LTD

Exploits on Organizations Worldwide Doubling every Two Hours after Microsoft's Revelation of Four Zero-days

MARCH 11, 2021

ASPEN MESH

Top 9 Takeaways from IstioCon 2021

MARCH 11, 2021

dynatrace

Dynatrace Managed release notes version 1.212

MARCH 11, 2021

MARCH 11, 2021

mongoDB.

Optimize Data Modeling and Schema Design with Hackolade and MongoDB

MARCH 11, 2021

CONFLUENT

Integrating Apache Kafka Clients with CNCF Jaeger at Funding Circle Using OpenTelemetry

MARCH 11, 2021

bridgecrew

Built-in Helm chart scanning with Checkov!

MARCH 11, 2021

harness

Chaos Engineering With Continuous Delivery

MARCH 11, 2021

SYNOPSYS®

Don’t let supply chain security risks poison your organization

MARCH 11, 2021

THE LINUX FOUNDATION

How open source communities are driving 5G’s future, even within a large government like the US

MARCH 11, 2021

New Relic.

Nerdlog Roundup: All Your Hosts, Clusters, and Apps in a Single View with New Relic Navigator

Five Years in the Making
MARCH 11, 2021

kasten by Veeam

5 Best Practices to Backup Kubernetes
MARCH 11, 2021

solo.io

Live Hoot Episode Recap – Istio Debugging
MARCH 11, 2021

pagerduty

What's New: Updates to Event Intelligence, Compliance and Reporting, and More! by Vera Chan
MARCH 11, 2021

THE LINUX FOUNDATION

Review of Four Hyperledger Libraries- Aries, Quilt, Ursa, and Transact
MARCH 11, 2021

circleci

Forbes names CircleCI to America's Best Startup Employers list
MARCH 11, 2021

THUNDRA

The CI/CD War of 2021: A Look at the Most Popular Technologies
MARCH 11, 2021

ebay

Shaping a Career Around Vintage Toy Organs
MARCH 11, 2021

MARCH 10, 2021

cloudbees

Major Security and Accessibility Updates in CloudBees CI and Jenkins

MARCH 10, 2021

SENTRY

Performance Monitoring Support for React Native

MARCH 10, 2021

Teleport

Preventing CSRF Attacks

MARCH 10, 2021

Cockroach Labs

When (& Why) You Should Use Change Data Capture

MARCH 10, 2021

Cribl

How to Architect Your LogStream to LogStream Data Flows

MARCH 10, 2021

SAP

Vulnerability data about open-source software should be open too!

MARCH 10, 2021

Hewlett Packard Enterprise

Data intelligence is the new oil

MARCH 10, 2021

MARCH 10, 2021

tetrate

Series B financing : Next step in Tetrate journey

MARCH 10, 2021

Tricentis

Streamline ServiceNow testing with a test automation strategy

MARCH 10, 2021

redislabs HOME OF REDIS

Redis 6.2—the “Community Edition”—Is Now Available

MARCH 09, 2021

StackPulse

How to Establish Your Service Level Objectives (SLOs)

MARCH 09, 2021

accurics

Infrastructure as Code Security Through Programmatic Controls

MARCH 09, 2021

styra

Linting Rego with... Rego!

MARCH 09, 2021

okta

Fast Java Made Easy with Quarkus and JHipster

MARCH 07, 2021

MARCH 02, 2021

fauna

Our Core Principles
MARCH 02, 2021

LaunchDarkly

Reflections on Black History Month at LaunchDarkly
FEBRUARY 25, 2021

ThousandEyes

ThousandEyes 2021 Predictions: Six Forecasts for the New Year
JANUARY 25, 2021

APPDYNAMICS

Why Is APM Important? Breaking Down the Benefits
JANUARY 25, 2021

WSO2

WSO2 Partner Awards 2020: Recognizing Our Extended Family for a Job Well Done
JANUARY 20, 2021

CLOUD FOUNDRY

Protecting Data In Your Cloud Foundry Applications (A Hands-on Lab Story)
JANUARY 13, 2021

CiTRIX

Tolly Group confirms Citrix ADC's performance leadership
JANUARY 12, 2021

AUGUST 27, 2020

vmware

Join us at our new blog home

JULY 02, 2020

ARCHITECTURE

Cloud Native

Containers

Edge/IoT

Microservices

Networking

Serverless

Storage

DEVELOPMENT

Cloud Services

Data

Development

Machine Learning

Security

OPERATIONS

CI/CD

Culture

DevOps

Kubernetes

Monitoring

Service Mesh