EdgeFaaS Collaborative R&D for Open Edge Scenarios

Arm & China Unicom

Tina Tsou     Mingxuan Li
Arm           China Unicom
EdgeFaaS Collaborative R&D for Open Edge Scenarios SoW

Pod1
- Triton
- EdgeFaaS
- Parsec Client

Pod2
- Triton
- EdgeFaaS
- Parsec Client

K3S Server

K3S Agent

SONiC

Parsec Service

ASIC drivers
Network drivers
Platform drivers
ARM
EdgeFaaS Collaborative R&D for Open Edge Scenarios SoW

• Task1: Integrate K3S
  - 1.1 K3S porting on SONiC platform.
  - 1.2 Run K3S server and K3S client on different SONiC devices.
  - 1.3 Create and run 2 Pods with K3S on one SONiC device.

• Task2: Integrate Parsec
  - 2.1 Check in parsec code for Akraino SW release5.(CI/CD)
  - 2.2 Port parsec to SONiC platform
  - 2.3 Run parsec service directly on SONiC.
  - 2.4 Run parsec client on a K3S pod, and make parsec client communicate success with parsec service.
  - 2.5 Run multi parsec clients on K3S pods, and make parsec clients communicate success with a same parsec service.

• Task3: Integrate EdgeFaaS
  - 3.1 Check in EdgeFaas code for Akraino.(CI/CD)
  - 3.2 Port edgeFaaS to SONiC platform.
  - 3.2 Run two edgeFaaS functions on different K3S pods, and make the edgeFaaS functions communicate success.

• Task4: Integrate Triton
  - 4.1 Check in Triton code for Akraino release6.(CI/CD)
  - 4.2 Port Triton to SONiC platform.
  - 4.3 Run two triton test case on different K3S pods.
K3S

K3s User
kubect! get pods

K3s Server

Server Node

Server Node

Server Node

K3s Agents
also called worker nodes

Agent Node

Agent Node

Agent Node

Load Balancer

External Traffic

Example configuration for nodes running your apps and services
Parsec is an open-source initiative to provide access to secure services and cryptographic operations in a platform-agnostic way, with a key goal being to make it easy for developers to build and run secure applications. Parsec runs as a service on an edge device, responding to client requests from applications running on the host that are posted to the transport endpoint by one of the language-native client libraries. These are key to Parsec's native support in various languages.
Parsec

Application Identity and Multi-Tenancy

Multiple Applications

Parsec Client Library

Identity

Proof

Authentication (pluggable)

Provider-Specific API/endpoint

Verification

Any Identity Provider

Secure Storage (HSM/TPM/TEE)

https://community.arm.com/developer/research/b/articles/posts/smarter-parsec-security-at-the-edge
Triton for Embedded Devices

- Triton is supported on Arm embedded platforms: Jetson Xavier, Jetson Nano, Raspberry Pi

Supported Features:

- TensorFlow 2.4, TensorFlow 1.15.5, TensorRT 7.1, ONNX RT 1.8, and Custom backends
- Model Ensembling
- CPU or GPU support
- Shared Library and C API
- C++ and Python client libraries and examples
China Unicom Use Case

- EdgeFaaS for Open Edge Scenarios
Mingxuan Li, Senior engineer of the Future Network Research Center of China Unicom Research Institute, mainly engaged in research on cloud native, cloud computing, computing power network, big data, and mobile Internet. I mainly track open source projects such as OpenStack, CNCF, LFN, and participate in ITU international standards, CCSA industry standards, etc. At present, I have published more than 20 papers, applied for more than 20 patents, more than 10 authorized patents, applied for 5 software copyrights, and jointly led the formulation of more than 10 industry standards.

E-mail: limx59@gmail.com
limx59@chinaunicom.cn
EdgeFaaS is to open edge computing based on "function as a service" capabilities, which is different from traditional MEP in:

- **Different calling ways**: Compared with PaaS in the application store, through function as a service, and through function calls to achieve capability opening.
- **Lightweight**: OpenFaaS and K3S are integrated to realize "Computing Power as a Service" at the edge.
- **Different billing models**: Realize computing power transactions according to the function call billing model, breaking the traditional virtual machine leasing method.
Scenarios Requirements

› Scenario 1: EdgeFaaS on SONiC
   › 1.1 K3S porting on SONiC platform.
   › 1.2 EdgeFaaS Open Edge Gateway API
   › 1.3 SONiC declares function API in SD-WAN network.

› Scenario 2: Computing Power Certification on Parsec at the Edge
   › 2.1 Edge Node implements certificate authentication
   › 2.2 Certificate preservation and anti-tampering based on Parsec
   › 2.3 Edge Node is registered to K3S with certificate

› Scenario 3: Computing power service based on EdgeFaas
   › 3.1 Realize the function call of edge computing power based on EdgeFaaS.
   › 3.2 Realization of billing for function operation based on license.
   › 3.3 Realize cloud-side collaboration based on EdgeFaaS.

› Scenario 4: SDC Smart Monitor based on Triton
   › 4.1 Realize AI algorithm based on Triton running on SDC
   › 4.2 Realization of GPU on the Nvidia nano sharing process based on Triton
Overall Technical Architecture

1. Authorization: Token is synchronized to the certification center, and the certificate is issued through the certification center.

2. Certificate issuance: The certificate authority issues the certificate under the Token and saves it to the TEE in the Edge Node.

3. Registration: Edge Node extracts the Token in the certificate and registers it in Edge Equipments equipment, and becomes the computing node of the edge cluster.

4. Certification process: The certificate is issued through the certification center.

Develop Platform

Request

Routing

SD-WAN

- FaaS statement
- Service routing
- Service registration
- Service discovery
- Signal Processing

Open Edge Network Equipment (Edge Gateway)

- SRv6
- BGP News release
- Framework
- Service process management
- ......
SONiC is an open source network operating system based on Linux that runs on switches from multiple vendors and ASICs. SONiC offers a full-suite of network functionality, like BGP and RDMA, that has been production-hardened in the data centers of some of the largest cloud-service providers. SONiC offers teams the flexibility to create the network solutions they need while leveraging the collective strength of a large ecosystem and community.

https://github.com/Azure/SONiC
Parsec Certification
The Arm NN TFLite Delegate can plug directly into the TFLite Runtime

- Greater flexibility for Android developers over NNAPI
- Arm specific CPU and GPU optimizations accessible through Arm NN
- All TFLite models can be supported:
  - Key operators accelerated through Arm NN and ACL
  - Unsupported operators processed through TFLite CPU Ref
- Also suitable for non-Android environments

Object Detection based on TFLite running on Raspberry Pi
- Recognition rate: 66%
Implementation

- EdgeFaaS on SONiC networking test
Test Network Topology

Router
192.168.0.1
255.255.255.0

Sonic
192.168.0.116
admin/admin

NVIDIA Nano
192.168.0.118
nano2g/admin

PC
192.168.0.104

Demo1: NVIDIA Jetson Nano #1 joins network
Demo1: NVIDIA Jetson Nano #1 joins network

NVIDIA Jetson Nano #1

Authentication Agent
K3S Agent
Parsec Client
Parsec Server

SONIC Gateway

K3S Server
Parsec Client
Parsec Server
Authentication Server

While(1)

Check node list
find a new k3s agent,
got agent info(ip...)

Call parsec client to encrypt the raw data with parsec server public key

While(1)

find a new k3s agent,
got agent info(ip...)

Call parsec client to encrypt the raw data with parsec server public key

trigger authentication flow(use IP:port), with server IP, send a raw text to Authentication Agent.

Call parsec client to encrypt the raw data with parsec server public key

push encryptText to authentication server

decrypt encryptText through parsec client

get plaintext

if failed --> disconnect NVIDIA Jetson Nano #1, remove from k3s server node list

Demo2: Get cluster system info

1. Web Browser
2. EdgeFaas
3. SONiC Gateway
4. Win/Ubuntu

Pod1
- EdgeFaas
- Parsec Server
- K3S Server
- SONiC

Pod2
- Parsec Client
- Authentication server

Pod3
- Parsec Client
- Triton client
- Ubuntu

Pod4
- Triton server
- Authentication Agent
- Parsec Server
- K3S Agent

Pod5
- Triton server
- Authentication Agent
- Parsec Server
- K3S Agent

Pod6
- Triton client
- Ubuntu

NVIDIA Jetson Nano #1
- Pod5

NVIDIA Jetson Nano #2
- Pod6

CAM

Demo2: Get cluster system info

PC

Web browser

http://xx.xx.xxx/funca

Return web info

SONIC Gateway

EdgeFaas FuncA.py

Authentication Server

K3s server

curl -v -X GET 127.0.0.1:8301/nodes
curl -v -X GET 127.0.0.1:8301/pods

curl kubectl get nodes -o json
kubectl get pods -o json

Return json

Return web info
Demo3: Image recognition

- Win/Ubuntu PC
- SONiC Gateway
  - Parsec Server
  - Authentication server
- Pod1
  - EdgeFaas
  - FuncA.py
  - FuncB.py
  - FuncC.py
- Pod2
  - EdgeFaas
  - Parsec Client
- Pod3
  - NVIDIA Jetson Nano #1
  - Triton server
  - Triton client
  - K3S Agent
  - Ubuntu (JetPack)
- Pod4
  - NVIDIA Jetson Nano #2
  - Triton server
  - Triton client
  - K3S Agent
  - Ubuntu (JetPack)
- Pod5
- Pod6
- Web browser
- EdgeFaas web api
- Parsec Server
Demo3: Image recognition

PC

Web browser

http://xx.xx.xxx/funcb

SONIC Gateway

EdgeFaas FuncB.py

Send image data

Triton Client

Request image analysis

Triton Server

Return image catalog

NVIDIA Jetson Nano #x

Return image catalog

Return web info

Demo 4: Get Camera Image

1. Web Browser
2. SONiC Gateway
3. Pod 3
4. Pod 4
5. Pod 5
6. Pod 6

Pods:
- Pod 1: Win/Ubuntu
- Pod 2: Parsec Server
- Pod 3: NVIDIA Jetson Nano #1
- Pod 4: NVIDIA Jetson Nano #2
- Pod 5: Triton server
- Pod 6: Triton server

Services:
- SONiC
- EdgeFaas
- FuncA.py
- FuncB.py
- FuncC.py

Agents:
- K3S Agent
- K3S Server
- Triton Client
- Triton Server
- Camera Client
- Triton Server
- Triton Server

Software:
- Ubuntu (JetPack)
- Triton Server
- Parsec Server
- Authentication Agent
- Web API
- Authentication Server
- Web API
- Web API
- Web API

Devices:
- Camera
- Win/Ubuntu PC
- EdgeFaas web app

Demo 4: Get Camera Image

PC

Web browser

http://xx.xx.xxx/funcc

SONIC Gateway

EdgeFaas FuncC.py

Send request

NVIDIA Jetson Nano #x

Camera Client

Camera device

Request Capture image

Return image

Return web info

Return image
Demo5: Update new Function

Pod1
- EdgeFaas
  - FuncA.py
  - FuncB.py
  - FuncC.py
- Parsec Server
- SONiC Gateway
- Web Browser
- Win/Ubuntu

Pod2
- Parsec Client
- Authentication server

Pod3
- Parsec Client
- Triton client
- Authentication Agent

Pod4
- Parsec Server
- Triton server

Pod5
- Parsec Client
- Authentication Agent

Pod6
- Triton client
- Triton server

Win/Ubuntu PC
- EdgeFaas, web api

CAM
Demo5: Update new Function

Demo6: Get API list

1. EdgeFaaS web api

PC

Win/Ubuntu

Web Browser

SONiC Gateway

SONiC

Parsec Server

Authentication server

Pod1

Pod2

1

2

EdgeFaaS web api

Pod3

Pod4

Pod5

Pod6

NVIDIA Jetson Nano #1

NVIDIA Jetson Nano #2

Ubuntu(JetPack)

Ubuntu(JetPack)

Triton server

Triton server

Parsec Server

Parsec Server

Triton client

Triton client

Authentication Agent

Authentication Agent

Parsec Client

Parsec Client

Web Browser

Authentication server

FuncA.py

FuncB.py

FuncC.py

Triton client

Triton client

Triton server

Triton server

Demo 6: Get API list

PC

Web browser

http://xx.xx.xxx/mgmt

SONiC Gateway

EdgeFaas

Return list
Thinking of Application

➢ EdgeFaaS on SONiC in Smart Manufacturing
**Spatiotemporal splicing positioning technology**

Realize the integration of hierarchical computing, intelligent computing power and quality traceability

**Production process:** Based on 5G communication, deep reinforcement learning and other technologies, open up the existing final assembly production management, use the **area array recognition** and **spatiotemporal splicing positioning technology** of image stitching, and standardize the processing algorithm of the shooting results to realize the segmented construction and overall assembly of components in the mobile environment process.

The re-rendering and less-changing content is placed in the edge computing power processing, and the light-rendering and multi-changing content is placed in the end computing power processing, so as to realize the low-latency processing of rendering and rendering after the panels are spliced.