Smart Citites R6 Installation Guide

Introduction

Smart Cities is an edge resource scheduling solution.

This guide setup a cluster network, one switch run with k3s server , one Nvidia Nona run with k3s node.

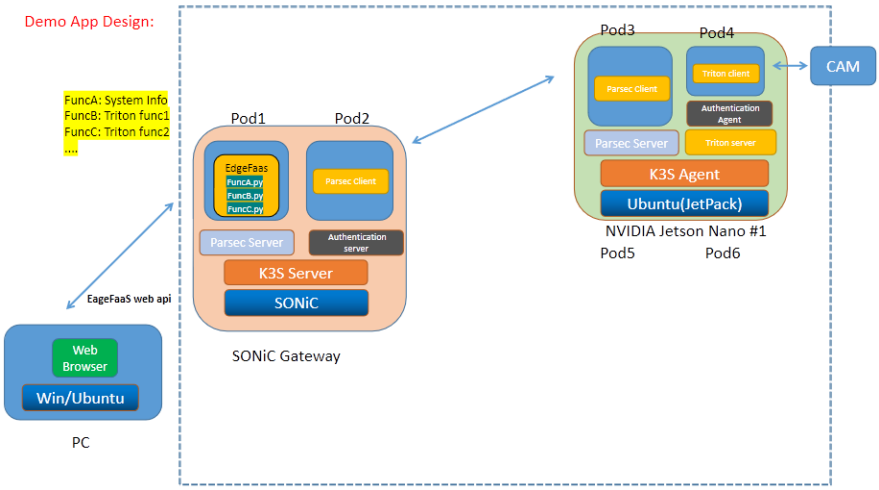
EdgeFaaS provide API to access the cluster network's capability.

License

How to use this document

The document describes how to compile demo app from source code, and deploy to device.

Deployment Architecture



- Pre-Installation Requirements
  - **Deploy Device Requirements**
    - Network Switch run with SONiC
      - CPU core: 4
      - RAM: 4G
      - HDD: 16G
    - Nvidia Nano run with Jetson
      - CPU core: 4
      - RAM: 2G
      - HDD: 32G
  - **Complie Device Requirements**
    - Arm64 device run with Ubunut 18.04+
    - CPU core: 4
    - RAM: 4G
    - HDD: 32G
    - Rust 1.54.0 +
    - Docker 20.10.1+
    - Golang 1.16+

- Build PARSEC

  $ git clone git clone "https://gerrit.akraino.org/r/a/cassini"

  $ cd cassini/smartcities/parsec

  $ cargo build --release --features "mbed-crypto-provider,direct-authenticator"

  The compiled binary executable is in ./target/release/parsec.

  Template of config file is in ./config.toml, it will be modify when deloy.

  Template of parsec.service file is in ./systemd-daemon/parsec.service it will be modify when deloy.

- Build Demo App
  - **Build ParsecClient**

    $ cd cassini/smartcities/ParsecClient/

    $ ./build.sh

    $ sudo docker images

    REPOSITORY TAG IMAGE ID CREATED SIZE

    parsec-client v1.0 aef2a010b6b5 3 months ago 17.3MB

    $ sudo docker save aef2a010b6b5 > parsec-client-v1.0-docker-aarch64.tar

    The saved parsec-client-v1.0-docker-aarch64.tar will use to delpoy on SONiC and Nvidia Nona.

- **Build NodeAuthServer**

  $ cd cassini/smartcities/NodeAuthServer/

  $ go build
- **Build NodeAuthAgent**

  $ cd cassini/smartcities/NodeAuthAgent/

  $ go build
- **Build CameraClient**

  $ cd cassini/smartcities/CameraClient/

  $ go build
- **Build TritonClient**

  $ cd cassini/smartcities/TritonClient/

  $ ./build.sh

  $ sudo docker images

  REPOSITORY TAG IMAGE ID CREATED SIZE

  `triton`-client v1.0 deaf4b1027ed 3 weeks ago 1.32 GB

  `$ sudo docker save deaf4b1027ed > triton-client-v1.0-docker-aarch64.tar`

  The saved triton-client-v1.0-docker-aarch64.tar will use to delpoy on Nvidia Nona.

- Deploy Demo App on SONiC
  - **Connect to SONiC**

    Use serial console connect to SONiC device, serial param is below:

    Baud rate: 115200

    Data bit: 8

    Parity: None

    Stop bits: 1

    Contorol: None
  - **Deploy Parsec server**
    1. **add system user parsec (NEED login by root)**

       $ sudo useradd -m parsec

       $ sudo passwd parsec

       $ sudo usermod -s /bin/bash parsec

       $ sudo groupadd parsec-clients
    2. **create necessary directorys (NEED login by root)**

       $ sudo mkdir /var/lib/parsec

       $ sudo chown parsec:parsec /var/lib/parsec

       $ sudo chmod 700 /var/lib/parsec

       $ sudo mkdir /etc/parsec

       $ sudo chown parsec:parsec /etc/parsec

       $ sudo chmod 700 /etc/parsec

       $ sudo mkdir /usr/libexec/parsec

       $ sudo chown parsec:parsec /usr/libexec/parsec

       $ sudo chmod 700 /usr/libexec/parsec

       $ sudo mkdir /home/parsec/run/

$ sudo chown parsec:parsec-clients /home/parsec/run/

$ sudo chmod 750 /home/parsec/run/

3. **deploy files (NEED login by parsec)**

   $ cd ~

   $ pwd

   /home/parsec

   $ cp xxx/parsec/target/release/parsec /usr/libexec/parsec

   $ chmod +x /usr/libexec/parsec/parsec

   $ cp xxx/parsec/config.toml /etc/parsec/config.toml

   Note: need unmark **allow_root = true** in config.toml.
4. **add service for systemctl (NEED login by root)**

   $ sudo vim /etc/systemd/system/parsec.service
   [Unit]
   Description=Parsec Service
   Documentation=https://parallaxsecond.github.io/parsec-book/parsec_service/install_parsec_linux.html

   [Service]
   WorkingDirectory=/home/parsec/
   ExecStart=/usr/libexec/parsec/parsec --config /etc/parsec/config.toml

   [Install]
   WantedBy=multi-user.target
5. **start parsec servive (NEED login by root)**

   $ sudo systemctl enable parsec.service

   $ sudo systemctl start parsec.service

- ## Deploy Parsec Client
  1. import docker image.
     $ sudo docker load < parsec-client-v1.0-docker-aarch64.tar

     $ sudo docker images

     $ sudo docker tag d396f7adeed3 parsec-client:v1.0
  2. run image.
     $ sudo docker run --restart=always -d -p8300:8300 -v /home/parsec/run:/run/parsec parsec-client:v1.0

     $ curl -v -d '{"Name": "GoClient"}' 127.0.0.1:8300/client
  3. create key pairs and export the public key.
     $ curl -v -d '{"Name": "GoClient"}' 127.0.0.1:8300/client

     $ curl -v -d '{"Name": "GoClient", "KeyName": "MyEncKey"}' 127.0.0.1:8300/keyenc

     $ curl -v -X GET -d '{"Name": "GoClient", "KeyName": "MyEncKey"}' 127.0.0.1:8300/key

- ## Deploy NodeAuthServer
  1. deploy files
     $ sudo mkdir /usr/libexec/NodeAuth

     $ sudo cp xxx/NodeAuthServer /usr/libexec/NodeAuth

     $ sudo chmod +x /usr/libexec/NodeAuth/NodeAuthServer
  2. add service for systemctl

     $ sudo vim /etc/systemd/system/NodeAuthServer.service
     [Unit]
     Description=Node Auth Server

     [Service]
     WorkingDirectory=/usr/libexec/NodeAuth
     ExecStart=/usr/libexec/NodeAuth/NodeAuthServer

     [Install]
     WantedBy=multi-user.target
  3. start parsec servive
     $ sudo systemctl enable NodeAuthServer.service

     $ sudo systemctl start NodeAuthServer.service

- ### Deploy k3s Server
  1. install and get token

     $ curl -sfL https://get.k3s.io | sh -s - --docker

     $ sudo cat /var/lib/rancher/k3s/server/node-token
     K10f3a81f8be38c4f230e45d330b153a6266a665d3310764d0d09075c2d4a40aa5b::server:
     a3abba7155f2f7b4684dbc548724ed22
- ### Deploy EdgeFaas
  1. run by docker

     $ sudo docker run --name edgefaas --restart=always -d -p8888:8080 registry.gitlab.com/arm-research/smarter/edgefaas
     /edgefaas:v1-1-0
  2. install demo apis

     $ curl 192.168.0.118:8888/mgmt -d "http://192.168.0.104:8080/sysinfo.py"

     $ curl 192.168.0.118:8888/mgmt -d "http://192.168.0.104:8080/camera.py"

     $ curl 192.168.0.118:8888/mgmt -d "http://192.168.0.104:8080/image.py"

     Note: 192.168.0.118 is EdgeFaas device, 192.168.0.104 is PC.

- Deploy Demo App on Nvidia Nano
  - ### Deploy Parsec Server
    1. **prepared files directory structure**

       parsec-server-aarch64/

       parsec

       config.toml

       parsec.service
    2. **add system user parsec (NEED login by root)**

       $ sudo useradd -m parsec

       $ sudo passwd parsec

       $ sudo usermod -s /bin/bash parsec

       $ sudo loginctl enable-linger parsec

       $ sudo groupadd parsec-clients
    3. **create necessary directorys (NEED login by root)**

       $ sudo mkdir /var/lib/parsec

       $ sudo chown parsec:parsec /var/lib/parsec

       $ sudo chmod 700 /var/lib/parsec

       $ sudo mkdir /etc/parsec

       $ sudo chown parsec:parsec /etc/parsec

       $ sudo chmod 700 /etc/parsec

       $ sudo mkdir /usr/libexec/parsec

       $ sudo chown parsec:parsec /usr/libexec/parsec

       $ sudo chmod 700 /usr/libexec/parsec

       $ sudo mkdir /home/parsec/run/

       $ sudo chown parsec:parsec-clients /home/parsec/run/

       $ sudo chmod 750 /home/parsec/run/
    4. **deploy files (NEED login by parsec)**

       $ cp xxx/parsec-server-aarch64/parsec /usr/libexec/parsec

       $ chmod +x /usr/libexec/parsec/parsec

       $ cp xxx/parsec-server-aarch64/config.toml /etc/parsec/config.toml

$ mkdir -p ~/.config/systemd/user

$ cp xxx/parsec-server-aarch64/parsec.service ~/.config/systemd/user

5. **start parsec servive (NEED login by parsec)**

$ systemctl --user enable parsec

$ systemctl --user start parsec

Deploy Parsec client

- import docker image
  $ sudo docker load < parsec-client-v1.0-docker-aarch64.tar
  $ sudo docker images
  REPOSITORY TAG IMAGE ID CREATED SIZE
  <none> <none> d396f7adeed3 2 months ago 16.9MB
  $ sudo docker tag d396f7adeed3 parsec-client:v1.0
- run image
  $ sudo docker run --restart=always -d -p8300:8300 -v /home/parsec/run:/run/parsec parsec-client:v1.0
  $ curl -v -d '{"Name": "GoClient"}' 127.0.0.1:8300/client
  ...
  HTTP/1.1 200 OK
  ...
- import public key
  $ curl -v -d '{"Name": "GoClient", "KeyName": "MyPubKey", "Message":"ssh-rsa
  MIIBCgKCAQEAui7zi+ehpbtqa+haTF2S7dB01QKf2zpIpy
  //58x9j4L1ZR44h7ftdymJlg3c07VBRVEx5BUIuaXPZ2g4Yq5Un21LK9GQoPvisIU53ePQN5anY4sO+NhqPOnaobkX02Pflp1m5
  EYjyCgVY62of/DXaPyr91xFeeyvRJFgfyPA+xk7VCsbjtNEL118IhJLmMVqNq/OgDDTvCN93hgJD5D
  /nRvXhe4CVR6ZYvBLF1E7blwwcq3EcmtYO+u3l0mPQSm9OR6YLhqxTcw29G/MBP+1X6yokALE
  /0Ykt0FxIOIQDkdukKtK50p38kzluQ9iHIOhG2QWhTpqwa7boQyeXcLiLQIDAQAB GoClient_MyEncKey"}' 127.0.0.1:8300/key

Deploy NodeAuthAgent

- deploy files
  $ sudo mkdir /usr/libexec/NodeAuth
  $ sudo cp xxx/NodeAuthAgent /usr/libexec/NodeAuth
  $ sudo chmod +x /usr/libexec/NodeAuth/NodeAuthAgent
- config systemctl service
  $ sudo vim /etc/systemd/system/NodeAuthAgent.service
  [Unit]
  Description=Node Auth Agent

  [Service]
  WorkingDirectory=/usr/libexec/NodeAuth
  ExecStart=/usr/libexec/NodeAuth/NodeAuthAgent

  [Install]
  WantedBy=multi-user.target
- start service
  $ sudo systemctl enable NodeAuthAgent.service
  $ sudo systemctl start NodeAuthAgent.service

Deploy k3s agent

- install with k3s server token
  $ curl -sfL https://get.k3s.io | K3S_URL=https://192.168.0.116:6443
  K3S_TOKEN=K10f3a81f8be38c4f230e45d330b153a6266a665d3310764d0d09075c2d4a40aa5b::server:
  a3abba7155f2f7b4684dbc548724ed22 sh -s - --docker

Deploy Triton Server

- get JetPack
  $ wget https://github.com/triton-inference-server/server/releases/download/v2.17.0/tritonserver2.17.0-jetpack4.6.tgz
  $ tar zxvf tritonserver2.17.0-jetpack4.6.tgz
- Installation dependent environment
  *$ sudo apt-get update && *
  *sudo apt-get install -y --no-install-recommends *
  *software-properties-common *
  *autoconf *
  *automake *
  *build-essential *
  *cmake *
  *git *
  *libb64-dev *
  *libre2-dev *
  *libssl-dev *
  *libtool *
  *libboost-dev *
  *libcurl4-openssl-dev *
  *libopenblas-dev *
  *rapidjson-dev *
  *patchelf *
  zlib1g-dev
  $ sudo apt remove cmake
  $ wget https://cmake.org/files/v3.21/cmake-3.21.0.tar.gz

$ tar -xf cmake-3.21.0.tar.gz
$ cd cmake-3.21.0 && ./configure && sudo make install
*$ sudo apt-get install -y --no-install-recommends *
*curl *
*pkg-config *
*python3 *
*python3-pip *
python3-dev
$ sudo pip3 install --upgrade wheel setuptools cython
$ sudo pip3 install --upgrade grpcio-tools numpy==1.19.4 future attrdict

- get Triton Model
$ git clone https://github.com/triton-inference-server/server
$ git checkout r21.12
$ cd docs/examples
$ ./fetch_models.sh
- test Triton server
$ cd tritonserver2.17.0
$ bin/tritonserver --model-repository=../model_repository --backend-directory=./backends --backend-config=tensorflow,version=2
$ curl -v localhost:8000/v2/health/ready
- creata systemctl service
$ sudo useradd -m triton
$ sudo passwd triton
$ sudo usermod -s /bin/bash triton
$ sudo loginctl enable-linger triton
$ cp -R /xxx/tritonserver2.17.0 ./
$ cp -R /home/u0u0/Documents/model_repository ./
$ mkdir -p ~/.config/systemd/user
$ vim ~/.config/systemd/user/triton.service
[Unit]
Description=Triton Server

[Service]
WorkingDirectory=/home/triton
ExecStart=/home/triton/tritonserver2.17.0/bin/tritonserver --model-repository=/home/triton/model_repository --backend-directory=/home/triton/tritonserver2.17.0/backends --backend-config=tensorflow,version=2

[Install]
WantedBy=default.target
- start service
$ systemctl --user enable triton
$ systemctl --user start triton

Deploy Triton client

- import docker image
$ sudo docker load < triton-client-v1.0-docker-aarch64.tar
$ sudo docker images
$ sudo docker tag a39dfgf7adeed3 triton-client:v1.0
- run image
$ sudo docker run --restart=always --add-host=host.docker.internal:host-gateway -d -p8302:8302 triton-client:v1.0

Maintenance

Blue Print Package Maintenance

Frequently Asked Questions

N/A

License

- References

N/A

- Definitions, acronyms and abbreviations

N/A