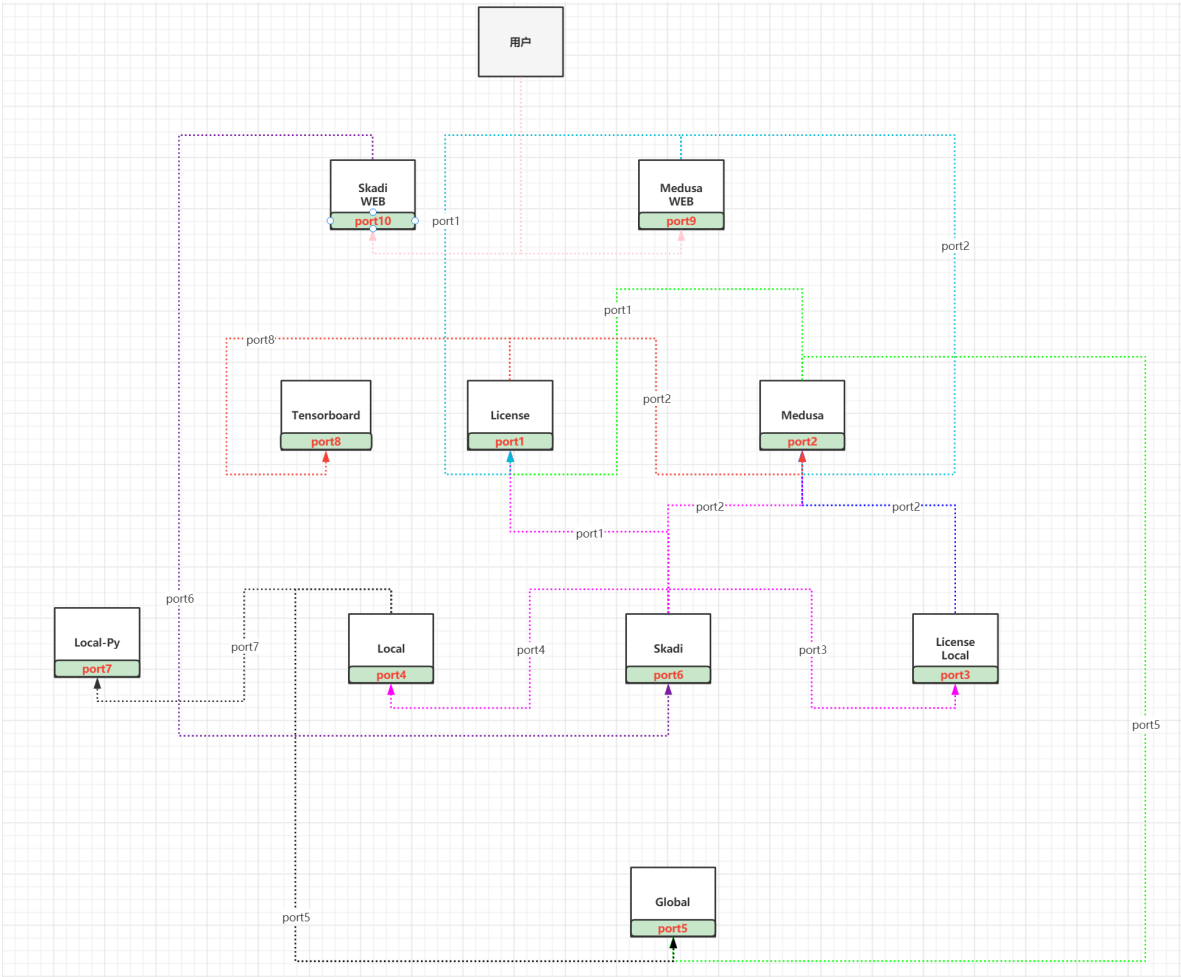
NV平台部署标准

一、部署模块预览



二、Medusa部署标准

1、License（Medusa）

1.1 .env

```
#容器名称
CONTAINER_NAME=medusa-license
#当前服务镜像
LICENSE_IMAGE=nvxhub.nvxclouds.net:9443/medusa/medusa_license:1012-1
#study服务IP或者hostname(一般license和study部署在同台服务器，这里也就可以写study的容器名通信)
STUDY_SERVER_IP=medusa-study
#study端口
STUDY_SERVER_PORT=10443
#废弃？
STUDY_SERVER_DOMAIN=192.168.10.28
#部署版本
VERSION=2.7.0
#资源调度
```

```
DATANODE_CPU_THRESHOLD=1
DATANODE_MEMORY_THRESHOLD=1
#废弃？
LICENSE_SERVER_IP=192.168.10.28
#license端口
LICENSE_SERVER_PORT=11443
#本地DEBUG端口配置,这个可以根据具体情况选择是否配置
LICENSE_SERVER_DEBUG_PORT=15005
#废弃？
DOMAIN=nvxclouds.net
#Tensorboard服务配置
TENSORBOARD_PUBLIC_DNS=192.168.10.100
TENSORBOARD_WEB_PORT=8200
TENSORBOARD_PORT_RANGE=8100-8200
#SGX 服务配置
GLOBAL_OLD_WEB_PORT=8080
GLOBAL_NEW_WEB_PORT=8080
GLOBAL_CNN_WEB_PORT=8080
GLOBAL_GWAS_WEB_PORT=8080
#计算默认端口
OLD_ANALYSIS_PORT=8081
NEW_ANALYSIS_PORT=8081
CNN_ANALYSIS_PORT=8081
BATCHED_LR_PORT=8081
#监管端服务配置
SUPERVISION_SERVER_IP=39.170.9.159
SUPERVISION_SERVER_PORT=10010
OAUTH2_IP=39.170.9.159
OAUTH2_PORT=9843
FILE_SERVER_IP=39.170.9.159
FILE_SERVER_PORT=9843
BLOCK_CHAIN_SERVER_IP=39.170.9.159:10010/api/v1/blockchain
#Docker 入口
DOCKER_ACCESS_HOST_ENDPOINT=172.17.0.1
#废弃？
#POSTGRES_PORT=54320
POSTGRES_VOLUMES_PATH=/home/database
#批量处理设置
BATCH_LOGISTIC_REGRESSION_SNP_PAGE_SIZE=500
BATCH_LOGISTIC_REGRESSION_SNP_BATCH_SIZE=1000
VERTIGO_SAMPLING_SIZE=1000
RANDOM_ID=OFF
VERTIGO_SAMPLING=OFF
#审批设置
APPROVAL_FILE_NAME=锘崴安全计算平台研究授权协议2020.docx
APPROVAL_FILE_TYPE=docx
APPROVAL_FILE_DIRECTORY=approvalFiles
UPLOADS_PATH=/home/nvx_dev/MedusaArtifacts/StudyServer/uploads
#用户系统资源显示信息设置，NONE 代表没有指定，逗号隔开同一客户的不同账号，分号隔开不同客户
CLIENT_ID=NONE
STUDY_METHODS=NONE
DATANODE_ID=NONE
#审批是否开启
APPROVAL=OFF
#安全模式是否开启
SECURE=ON
#SGX定期健康查询
SGX_HEALTH_CHECK=OFF
```

```
#审批白名单,逗号隔开用户ID
APPROVAL_WHITE_LIST=NONE
#是否开启BlockChain记录
BLOCKCHAIN=OFF
#期权交易
LIMIT_PRICE=1.0
ACCOUNT=00105687
PASSWORD=50798535
LOGIN_ID_MIN=5000000
LOGIN_ID_MAX=6000000
OPTION_REMOTE_SERVER=tcp://210.14.72.14:4400
ORDER_LIMIT=1
#DataBinning 文件Suffix
SUFFIX=0
#是否开启用户注册授权审批
AUTHORIZATION=OFF
SMS_NOTIFICATION=OFF
#数据库配置
MYSQL_HOST=mysql
#RabbitMQ
RABBIT_HOST=rabbitmq
```

1.2 docker-compose.yml

```
version: '3'

networks:
  localnet:
    external:
      name: localnet

services:
  medusa-license:
    image:  ${LICENSE_IMAGE}
    container_name: ${CONTAINER_NAME}
    networks:
      - localnet
    environment:
      RABBIT_HOST: ${RABBIT_HOST}
      MYSQL_HOST: ${MYSQL_HOST}
    env_file:
      - .env
    ports:
      - ${LICENSE_SERVER_PORT}:${LICENSE_SERVER_PORT}
      - ${LICENSE_SERVER_DEBUG_PORT}:${LICENSE_SERVER_DEBUG_PORT}
```

2、Study（Medusa）

2.1 .env

```
#容器名称
CONTAINER_NAME=medusa-study
#当前服务镜像
STUDY_IMAGE=nvxhub.nvxclouds.net:9443/skadi/medusaglobalserver:1025-1
#部署版本
VERSION=2.5.0
```

```
#数据库配置
MYSQL_HOST=mysql
#RabbitMQ
RABBIT_HOST=rabbitmq
#是否开启图像分类本地测试
DEBUG_MODE=OFF
#是否开启蚂蚁TEE本地测试
DEBUG_MAYI=ON
#废弃？
#STUDY_SERVER_IP=teststudy.nvxclouds.net
#废弃？
STUDY_SERVER_IP=medusa-study
#study服务端口
STUDY_SERVER_PORT=10443
#本地DEBUG端口配置,这个可以根据具体情况选择是否配置
STUDY_SERVER_DEBUG_PORT=15005
#license服务Ip、主机名、容器名
LICENSE_SERVER_IP=192.168.10.28
#license服务端口
LICENSE_SERVER_PORT=11443
#废弃？
STUDY_SERVER_DOMAIN=192.168.10.28
#资源调度
DATANODE_CPU_THRESHOLD=1
DATANODE_MEMORY_THRESHOLD=1
#废弃？
DOMAIN=nvxclouds.net
#Tensorboard服务配置
TENSORBOARD_PUBLIC_DNS=cnnresults.nvxclouds.net
TENSORBOARD_PUBLIC_IP=192.168.10.100
TENSORBOARD_WEB_PORT=8200
TENSORBOARD_PORT_RANGE=8100-8200
#SGX 服务配置
GLOBAL_CNN_WEB_PORT=8840
GLOBAL_GWAS_WEB_PORT=8840
#GLOBAL适用于原来历史研究
GLOBAL_OLD_WEB_PORT=8840
GLOBAL_NEW_WEB_PORT=8840
#GLOBAL适用于图像分类研究
GLOBAL_IMAGE_CLASSIFICATION_WEB_PORT=2080
GLOBAL_IMAGE_CLASSIFICATION_WEB_IP=192.168.10.31
#GLOBAL适用于DEEPSEG研究
GLOBAL_DEEPSEG_WEB_IP=192.168.10.31
GLOBAL_DEEPSEG_WEB_PORT=18080
#ANALYSIS适用于原来历史研究的
OLD_ANALYSIS_PORT=8741
NEW_ANALYSIS_PORT=8741
CNN_ANALYSIS_PORT=8741
BATCHED_LR_PORT=8741
#ANALYSIS适用于图像分类研究
IMAGE_CLASSIFICATION_ANALYSIS_PORT=2081
#ANALYSIS适用于DEEPSEG研究
DEEPSEG_ANALYSIS_PORT=18081
#监管端服务配置
SUPERVISION_SERVER_IP=39.170.9.159
SUPERVISION_SERVER_PORT=10010
OAUTH2_IP=39.170.9.159
OAUTH2_PORT=9843
```

```
FILE_SERVER_IP=39.170.9.159
FILE_SERVER_PORT=9843
BLOCK_CHAIN_SERVER_IP=39.170.9.159:10010/api/v1/blockchain
#Docker 入口
DOCKER_ACCESS_HOST_ENDPOINT=172.17.0.1
#废弃?
#POSTGRES_PORT=54320
POSTGRES_VOLUMES_PATH=/home/database
#批量处理设置
BATCH_LOGISTIC_REGRESSION_SNP_PAGE_SIZE=500
BATCH_LOGISTIC_REGRESSION_SNP_BATCH_SIZE=1000
VERTIGO_SAMPLING_SIZE=1000
RANDOM_ID=OFF
VERTIGO_SAMPLING=OFF
#审批设置
APPROVAL_FILE_NAME=锘崴安全计算平台研究授权协议2020.docx
APPROVAL_FILE_TYPE=docx
APPROVAL_FILE_DIRECTORY=approvalFiles
UPLOADS_PATH=/home/nvx_dev/MedusaArtifacts/StudyServer/uploads
#用户系统资源显示信息设置，NONE 代表没有指定，逗号隔开同一客户的不同账号，分号隔开不同客户
CLIENT_ID=NONE
STUDY_METHODS=NONE
DATANODE_ID=NONE
#审批是否开启
APPROVAL=OFF
#安全模式是否开启
SECURE=ON
#SGX定期健康查询
SGX_HEALTH_CHECK=OFF
#审批白名单,逗号隔开用户ID
APPROVAL_WHITE_LIST=NONE
#是否开启BlockChain记录
BLOCKCHAIN=OFF
#期权交易
LIMIT_PRICE=1.0
ACCOUNT=00105687
PASSWORD=50798535
LOGIN_ID_MIN=5000000
LOGIN_ID_MAX=6000000
OPTION_REMOTE_SERVER=tcp://210.14.72.14:4400
ORDER_LIMIT=1
#DataBinning 文件Suffix
SUFFIX=0
#是否开启用户注册授权审批
AUTHORIZATION=OFF
SMS_NOTIFICATION=OFF
#容器用户
USER=root
#volumes
ORDERS_PATH=./orders
TMP_PATH=/tmp
HOSTS_PATH=/etc/hosts
```

2.2 docker-compose.yml

```
version: '3'
```

```
networks:
  localnet:
    external:
      name: localnet

services:
  medusa-study:
    image: ${STUDY_IMAGE}
    container_name: ${CONTAINER_NAME}
    user: ${USER}
    env_file:
      - .env
    ports:
      - ${STUDY_SERVER_PORT}:${STUDY_SERVER_PORT}
      - ${STUDY_SERVER_DEBUG_PORT}:${STUDY_SERVER_DEBUG_PORT}
    volumes:
      - ${ORDERS_PATH}:/medusa_study/OptionAPIs/orders
      - ${TMP_PATH}:/tmp
      - ${HOSTS_PATH}:/etc/hosts
    networks:
      - localnet
    restart: always
```

3、Web（Medusa）

3.1 .env

```
#容器名称
CONTAINER_NAME=medusa-web
#页面镜像
WEB_IMAGE=nvxhub.nvxclouds.net:9443/skadi/medusa-web:20211025-2
#Nginx Https端口
HTTPS_PORT=8443
#Nginx Http端口
HTTP_PORT=8080
#容器用户
USER=root
##volumes
HOSTS_PATH=/etc/hosts
```

3.2 docker-compose.yml

```
version: '3'

networks:
  localnet:
    external:
      name: localnet

services:
 medusa-web:
   image: ${WEB_IMAGE}
   container_name: ${CONTAINER_NAME}
   user: ${USER}
   env_file:
     - .env
```

```
    ports:
      - ${HTTPS_PORT}:443
      - ${HTTP_PORT}:80
    volumes:
      - ${HOSTS_PATH}:/etc/hosts
    networks:
      - localnet
    restart: always
```

三、Skadi部署标准（本地计算端）

1、DataNode（Skadi）

1.1 .env

```
#容器名称
CONTAINER_NAME=skadi-datanode
#当前服务镜像名称
DATANODE_IMAGE=nvxhub.nvxclouds.net:9443/skadi/skadi-datanode:20211015-3
DATANODE_PORT=52442
DATANODE_DEBUG_PORT=15005
#study服务地址
STUDY_SERVER=192.168.10.28:10443
#license服务地址
LICENSE_SERVER=192.168.10.28:11443
#study服务IP
STUDY_SERVER_IP=192.168.10.28
#study服务端口
STUDY_SERVER_PORT=10443
#license服务IP
LICENSE_SERVER_IP=192.168.10.28
#license服务端口
LICENSE_SERVER_PORT=11443
#本地部署Global
SGX_SERVER_IP=192.168.10.30
SGX_SERVER_PORT=8081
#阿里云Global
SGX_SERVER_IP=101.37.255.96
SGX_SERVER_PORT=8841
WDT_SERVER_IP=192.168.10.30
SUPERVISION_SERVER=39.170.9.159:9843
BLOCKCHAIN_SERVER=39.170.9.159:9843
#配置local计算服务
#1、原来
LOCAL_COMPUTATION_URL_OLD=https://client-old:8082
LOCAL_COMPUTATION_URL_NEW=https://client-new:8083
#2、DeepSeg 请修改对应docker服务名字和端口
LOCAL_COMPUTATION_URL_TORCH=https://client-torch-depseg2:8081
#3、ImageClassification  请修改对应docker服务名字和端口
LOCAL_COMPUTATION_URL_TORCH1=https://192.168.10.3:8080
#-------------------------------------------------------------------------
#----------------v2.6.0 added-----------------------
#本地数据节点
LOCAL_NODE=skadi-datanode2:52442
DOCKER_ACCESS_HOST_ENDPOINT=192.168.10.100
#数据库
```

```
MYSQL_HOST=192.168.10.28
MYSQL_DB_NAME=medusa_datanode
#Docker入口
DOCKER_ACCESS_HOST_ENDPOINT=172.17.0.1
#DOCKER_ACCESS_HOST_ENDPOINT=192.168.10.14
WDT_SERVER_PORT=8080
WDT_SERVER_SEND_PORT=22888
#数据节点配置
DATANODE_ADMIN_ID=1
DATANODE_ID=1
#本地授权服务端口 license-local的端口
LICENSE_LOCAL_PORT=14442
#数据库
POSTGRES_PORT=54322
POSTGRES_VOLUMES_PATH=/home/database
#文件路径mount
UPLOADS_PATH=./uploads
OUTPUTS_PATH=./outputs
LR_OUTPUTS_PATH=./lr_outputs
LOG_PATH=./logs
WDT_PATH=./wdt_bin
KEY_PATH=./keys
ORIGINAL_PATH=./original
PRETREATMENT_PATH=./pretreatment
HOSTS_PATH=/etc/hosts
VERSION=2.4.0
#是否开启日志
LOG_SERVICE=ON
#是否开启审批
APPROVAL=OFF
#是否开启BlockChain记录
BLOCKCHAIN=OFF
SUPERVISION=OFF
```

1.2 docker-compose.yml

```
version: "3.7"

networks:
  localnet:
    external:
      name: localnet

services:
  medusa_datanode3:
    image: ${DATANODE_IMAGE}
    container_name: ${CONTAINER_NAME}
    env_file:
     - .env
    volumes:
     - ${UPLOADS_PATH}:/medusa_datanode/uploads
     - ${OUTPUTS_PATH}:/medusa_datanode/outputs
     - ${PRETREATMENT_PATH}:/medusa_datanode/pretreatment
     - ${ORIGINAL_PATH}:/medusa_datanode/original
     - ${LOG_PATH}:/medusa_datanode/logs
     - ${WDT_PATH}:/medusa_datanode/wdt_bin
     - ${KEY_PATH}:/medusa_datanode/keys
```

```
      - ${HOSTS_PATH}:/etc/hosts
    ports:
      - ${DATANODE_PORT}:${DATANODE_PORT}
      - ${DATANODE_DEBUG_PORT}:${DATANODE_DEBUG_PORT}
    environment:
      MYSQL_HOST: ${MYSQL_HOST}
      MYSQL_DB_NAME: ${MYSQL_DB_NAME}
    cap_add:
      - SYS_PTRACE
    networks:
      - localnet
    restart: always
```

2、License-Local（Skadi）

2.1 .env

```
#容器名称
CONTAINER_NAME=license-local
#当前服务镜像名称
LICENSE_LOCAL_IMAGE=nvxhub.nvxclouds.net:9443/skadi/skadi-license:dev
#服务端口
LICENSE_LOCAL_SERVER_PORT=14443
#Medusa研究服务器
STUDY_SERVER_IP=192.168.10.28
STUDY_SERVER_PORT=10443
#许可和token路径
CREDENTIALS_PATH=./credentials
HOSTS_PATH=/etc/hosts
#Medusa授权服务器
LICENSE_SERVER_IP=192.168.10.28
LICENSE_SERVER_PORT=11443
#本地数据节点
LOCAL_NODE=192.168.10.100:52443
#datanode(skadi服务的地址)
DATANODE_HOST=192.168.10.100
#版本
VERSION=2.7.0
```

2.2 docker-compose.yml

```
version: "3.7"

networks:
  localnet:
    external:
      name: localnet

services:
  license_local:
    image: ${LICENSE_LOCAL_IMAGE}
    container_name: ${CONTAINER_NAME}
    env_file:
      - .env
    ports:
      - ${LICENSE_LOCAL_SERVER_PORT}:13443
```

```
    volumes:
      - ${CREDENTIALS_PATH}:/license-local
      - ${HOSTS_PATH}:/etc/hosts
    environment:
      DATANODE_HOST: ${DATANODE_HOST}
    networks:
      - localnet
    restart: always
```

3、Local（计算端）

3.1 适用历史研究方法

.env

```
#容器名称
CONTAINER_NAME=client-new
CLIENT_NEW_IMAGE=nvxhub.nvxclouds.net:9443/sgx/img/nvxcloudstechdatanodestatis:1e1182c8669ecc98a0e8babdc9f9baea8fbb5ea0
GRPC_IMAGE=nvxhub.nvxclouds.net:9443/skadi/migrate/grpcserver:3296e077f9300778eaa61c20342e235da761ab7a
CLIENT_NEW_PORT=8086
#Global服务IP
SGX_SERVER_IP=192.168.10.30
#Global ANALYSIS端口
SGX_SERVER_PORT=8741
#skadi端口 datanode服务端口
DATA_NODE_PORT=52446
#env
DSTATIS_HTTP_PORT=1100
DSTATIS_HTTPS_PORT=8083
DSTATIS_HOSTNAME=0.0.0.0
DSTATIS_FLAG_BYPASS_IAS_VERIFY=ON
#GRPC_HOSTNAME=grpc-server
GRPC_PORT=50051

#volumes
UPLOADS=/home/uploads
OUTPUTS=/home/outputs
LOGS=/home/logs
LR_OUTPUTS=/home/lr_outputs
KEYS=/home/keys

#grpc
GRPC_CONTAINER_NAME=grpc-server
NUM_WORKERS=2
```

docker-compose.yml

```
version: '3'

networks:
  localnet:
    external:
      name: localnet
```

```
services:
  client-new:
    image: ${CLIENT_NEW_IMAGE}
    container_name: ${CONTAINER_NAME}
    ports:
      - ${CLIENT_NEW_PORT}:8083
    environment:
      - DSTATIS_HTTP_PORT=${DSTATIS_HTTP_PORT}
      - DSTATIS_HTTPS_PORT=${DSTATIS_HTTPS_PORT}
      - DSTATIS_HOSTNAME=${DSTATIS_HOSTNAME}
      - DSGX_REMOTE_ATTESTATION_PORT=${SGX_SERVER_PORT}
      - DSGX_REMOTE_ATTESTATION_HOST=${SGX_SERVER_IP}
      - DSTATIS_FLAG_BYPASS_IAS_VERIFY=${DSTATIS_FLAG_BYPASS_IAS_VERIFY}
      - GRPC_HOSTNAME=${GRPC_CONTAINER_NAME}
      - GRPC_PORT=${GRPC_PORT}
    command: [ "./start.sh" ]
    volumes:
      - ${UPLOADS}:/home/uploads
      - ${OUTPUTS}:/home/outputs
      - ${LOGS}:/home/logs
      - ${LR_OUTPUTS}:/home/lr_outputs
      - ${KEYS}:/home/keys
    networks:
      - localnet
    restart: always

  grpc-server:
    image: ${GRPC_IMAGE}
    container_name: ${GRPC_CONTAINER_NAME}
    ports:
      - ${GRPC_PORT}:50051
    environment:
      - PORT=50051
      - NUM_WORKERS=${NUM_WORKERS}
    command: [ "python3.7", "-u", "server.py" ]
    volumes:
      - ${UPLOADS}:/home/grpc_server/uploads
    networks:
      - localnet
    restart: always
```

playbook.yml

```
- hosts: datanode_servers
  vars:
    project_name: skadi-computition
    workspace: /home/jenkins/jenkins_data/workspace/{{project_name}}/computition
    workspace_dest: /home/hz_dev/skadi-deploy/computition
    ip_from: 192.168.10.16
    docker_compose_file: '{{workspace_dest}}/docker-compose.yml'
    env_file: '{{workspace_dest}}/.env'
  remote_user: hz_dev
  tasks:
    - name: Copy file of docker-compose to remote server
      synchronize:
        src: '{{item.src}}'
```

```
        dest: '{{item.dest}}'
      with_items:
        - { src: '{{workspace}}/docker-compose.yml', dest:
'{{docker_compose_file}}' }
        - { src: '{{workspace}}/.env', dest: '{{env_file}}' }
      delegate_to: '{{ip_from}}'
    - name: Docker-compose running
      shell: cd {{workspace_dest}}  && docker-compose up -d
```

3.2 适用图像分类研究方法

docker-compose.yml

```
version: "3.7"
services:
    datanode-deeplearning-cpp:
      image: "nvxhub.nvxclouds.net:9443/ai/datanode-deeplearning-cpp-
stable:0.1.1"
      container_name: datanode-deeplearning-cpp
      environment:
        - DIMAGE_HTTP_PORT=12000
        - DIMAGE_HTTPS_PORT=2080
        - DIMAGE_HOSTNAME=0.0.0.0
        - DSGX_REMOTE_ATTESTATION_PORT=2081
        - DSGX_REMOTE_ATTESTATION_HOST=192.168.10.31
        - DIMAGE_FLAG_BYPASS_IAS_VERIFY=ON
        - DeepLearningHost=192.168.10.100:50156
        - DeepLearningDevice=cuda:0
      command: ["./start.sh"]
      tty: true
      volumes:
        - ./data/outputs:/home/outputs
        - ./data/logs:/home/logs
        - /home/ltguo/nvxclouds/DataNode_AI/:/home/DataNode/
      networks:
        - localnet
      restart: always
      ports:
        - 12000:12000
        - 2080:2080
networks:
  localnet:
    external:
      name: localnet
```

docker-py-run.yml

```
#!/bin/bash
CONTAINER_NAME=datanode-deeplearning-python
DOCKER_IMAGE=nvxhub.nvxclouds.net:9443/ai/datanode-deeplearning-python-
stable:0.1.1
GRPC_SERVER_PORT=50156
docker container stop ${CONTAINER_NAME}
docker container rm ${CONTAINER_NAME}
docker run \
        -v /home/ltguo/.cache/torch/:/root/.cache/torch/ \
        -v /home/shouzhi/deploy/datanode12443/uploads:/data \
```

```
        -p ${GRPC_SERVER_PORT}:50055 \
        --gpus '"device=0"' \
        --restart always \
        -it -d --name ${CONTAINER_NAME} \
        ${DOCKER_IMAGE} \
        bash -c "./start.sh"
```

3.3 适用DEEPSEG研究方法

startLocal.sh

```
#!/bin/bash
docker container stop client-torch-depseg
docker container rm client-torch-depseg
docker run --gpus device=1 \
        -v /home/shouzhi/deploy/datanode52443/uploads:/home/uploads \
        -v /home/shouzhi/deploy/datanode52443/outputs:/home/outputs \
        -v /home/shouzhi/deploy/datanode52443/logs:/home/logs \
        -e DIMAGE_HTTP_PORT=11000 \
        -e DIMAGE_HTTPS_PORT=8080 \
        -e DIMAGE_HOSTNAME=0.0.0.0 \
        -e DSGX_REMOTE_ATTESTATION_HOST=192.168.10.31 \
        -e DSGX_REMOTE_ATTESTATION_PORT=18081 \
        -e DIMAGE_FLAG_BYPASS_IAS_VERIFY=ON \
        --restart always \
        -it -d --name client-torch-depseg \
        --network localnet \
        nvxhub.nvxclouds.net:9443/skadi/migrate/nvxcloudstechdatanodeai:58-
upcert \
        bash -c "./start.sh"
```

四、Global部署标准

1、适用历史研究方法

docker-compose.yml

```
version: "3.7"
services:
    global-server-new:
      image:
"nvxhub.nvxclouds.net:9443/sgx/img/nvxcloudstechsgxserverhw:4070da4cd8c95ff8ee53
423d24ab29745cd5f77d"
      ports:
        - 8080-8090:8080-8090
        - 22888-22898:22888-22898

      container_name: global_2.7.1
      devices:
        - /dev/isgx:/dev/isgx
      environment:
        - PUBLIC_IP_ADDRESS=192.168.10.31
        - HTTPS_PORT=8080
        - ANALYSIS_DEFAULT_PORT=8081
        - ANALYSIS_LAST_PORT=8090
```

```
        - WDT_PORT=22888
        - WDT_LAST_PORT=22898
        - CENTRAL_API_URL=${CENTRAL_API_HOST}/results/complete/secure
        - CENTRAL_API_UPDATES_URL=${CENTRAL_API_HOST}/results/updates/secure
        - CENTRAL_API_REGISTER_URL=${CENTRAL_API_HOST}/sgx/register
        - CENTRAL_API_ERROR_URL=${CENTRAL_API_HOST}/errorHandling/fail
        -
CENTRAL_API_INTERMEDIATE=${CENTRAL_API_HOST}/studies/result/intermediate
        - CENTRAL_API_TIMEOUT=${CENTRAL_API_HOST}/errorHandling/fail
        - CENTRAL_API_POLLING=${CENTRAL_API_HOST}/studies/postactions/polling
        - CENTRAL_ACCESS_TOKEN=${CENTRAL_ACCESS_TOKEN}
        - CENTRAL_ERROR_TOKEN=${CENTRAL_ERROR_TOKEN}
        - CENTRAL_VERIFY_SKIP=1
        - POLLING_TIMEOUT=180
        - POLLING_REFRESH_TIME=10
        - NODE_NAME=杭州计算节点
        -
NODE_ENCLAVE_ID=a8196a010cd13aa2e42415d2651676a112798c606df155227d46b3dfcb1acb01

      volumes:
        - ./ID_files:/home/build/ID_files
        - ./sealed_dir:/home/build/sealed_dir
          #- /home/hz/lijiacheng/GlobalServeNew:/root/sgx
          #network_mode: 'host'
      command: ["./start.sh"]
      restart: always
```

.env

```
CENTRAL_API_HOST=https://192.168.10.28:10443
CENTRAL_ACCESS_TOKEN=NOVOVIVO-token-M1LG1xLV8*c4MqzQH+SVJtvy4KoLObJt
CENTRAL_ERROR_TOKEN=NOVOVIVO-token-I%5lA%T?eppOR*7^G#U2O7g9nuPcMSQy
```

2、适用图像分类研究方法

3、适用DEEPSEG研究方法

docker-compose.yml

```
version: "3.7"

networks:
  localnet:
    external:
      name: localnet

services:
    global-server-new-pre:
```

```
      image:
"nvxhub.nvxclouds.net:9443/sgx/img/nvxcloudstechsgxserverhw:11dcc382473f9e3cb993
28c8f115d0aa12ab3d20"
      devices:
        - /dev/isgx:/dev/isgx
      container_name: global-depseg
      ports:
        - 18080-18081:18080-18081
      environment:
        - LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/sgxsdk/sdk_libs
        - HTTPS_PORT=18080
        - ANALYSIS_DEFAULT_PORT=18081
        - CENTRAL_API_URL=https://192.168.10.28:10443/results/complete
        - CENTRAL_API_UPDATES_URL=https://192.168.10.28:10443/results/updates
        - CENTRAL_ACCESS_TOKEN=NOVOVIVO-token-M1LG1xLV8*c4MqzQH+SVJtvy4KoLObJt
        - CENTRAL_VERIFY_SKIP=1
        - NODE_NAME=CNN计算节点
      networks:
         - localnet
      command: ["./start.sh"]
      restart: always
```

五、常用中间件部署