# Secretflow安装理论篇

## secretflow安装

安装包：完整版/lite

lite不包含深度学习依赖库，体积小

安装方式：docker/pypi/src

docker run -it secretflow-registry.cn-hangzhou.cr.aliyuncs.com/secretflow-anolis8:latest

## secretflow部署 – 仿真模式

适合验证代码效果

ray集群：每个集群有一个主节点和0-n个从节点

单机仿真：

sf.init(parties=…, address=”local”)

集群仿真：

1. 一台机器上部署ray主节点，ray start --head --node-ip-address=”…” --port = “…” --resources=’{“alice”:16}’ --include-dashboard=False --disable-usage-stats
2. 另一台部署从节点，ray start --address=”主节点通信地址” –resources=… --include… --disable…
3. 在任意一台机子上执行sf.init代码
4. 创建spu：建立cluster\_def，其中port不能和ray的端口冲突

## sf生产模式

### 基础版

每个参与方都需要同时执行代码

每一方都是独立的ray集群

ray, secretflow proxy, spu的端口不能冲突

### 不需要配置那么多端口的版本

端口太多了？考虑kuscia。后续会有专门kuscia课程

使用secretnote可以在一个页面上执行多方代码，类似Jupyter Notebook

推荐使用docker运行secretnote

## FAQ

教程文档：[Kuscia 入门教程 —— 快速开始 | Kuscia v0.8.0b0 | 隐语 SecretFlow](https://www.secretflow.org.cn/zh-CN/docs/kuscia/v0.8.0b0/getting_started/quickstart_cn)

镜像下载慢：用阿里云镜像

网络报错：端口占用

Docker运行sf集群，注意：

1、建议使用host网络模式

2、一台机器上运行多个docker容器并且接入一个docker network

# 实操篇

## Docker安装secretnote

环境：docker desktop在wsl Ubuntu 22.04内启用

访问容器:

执行docker inspect xxx：

“Networks”:{

…

“Aliases”:[…]

…

“Gateway”:”…”,

“IPAddress”:”…”

…

}

选择aliases, gateway, ipaddress中的任何一个都可以通信

1、下载镜像：docker pull secretflow/secretnote:1.5.0-amd64

2、按照<https://github.com/secretflow/secretnote新建docker-compose.yml>

3、执行docker compose up -d

不加-d会导致本次执行的终端窗口报废，无法退出，不能执行其他命令

4、浏览器输入<http://172.25.172.56:8090/> <http://172.25.172.56:8092/>可以交互

5、打开上一条其中一个网页，在浏览器窗口右上角按+，名称写alice, 地址写alice:8888, 同理输入bob和bob8888

6、notebooks上面创建psi任务，输入相应代码，选择好执行代码的参与方

如果需要debug，点击圈R，然后刷新页面，就可以重新执行代码

7、点左边一栏文件附近的刷新按钮，就能看到psi的结果文件

# 附录

## docker-compose.yml

services:

alice:

image: 'secretflow/secretnote:1.5.0-amd64'

platform: linux/amd64

environment:

- SELF\_PARTY=alice

- ALL\_PARTIES=alice,bob

ports:

- 8090:8888

entrypoint: /root/scripts/start.sh

volumes:

- /root/scripts

bob:

image: 'secretflow/secretnote:1.5.0-amd64'

platform: linux/amd64

environment:

- SELF\_PARTY=bob

- ALL\_PARTIES=alice,bob

ports:

- 8092:8888

entrypoint: /root/scripts/start.sh

volumes:

- /root/scripts

## psi(web)

一共四块代码，按顺序整理如下

### alice端

import secretflow as sf

cluster\_config={

"parties":{

"alice":{

"address":"alice:7889",

"listen\_addr":"0.0.0.0:7889"

},

"bob":{

"address":"bob:7900",

"listen\_addr":"0.0.0.0:7900"

}

},

"self\_party":"alice"

}

sf.shutdown()

sf.init(address="local", cluster\_config=cluster\_config)

### bob端

import secretflow as sf

cluster\_config={

"parties":{

"alice":{

"address":"alice:7889",

"listen\_addr":"0.0.0.0:7889"

},

"bob":{

"address":"bob:7900",

"listen\_addr":"0.0.0.0:7900"

}

},

"self\_party":"bob"

}

sf.shutdown()

sf.init(address="local", cluster\_config=cluster\_config)

### ab一起

import spu

import secretflow as sf

cluster\_def={

"nodes":[

{

"party":"alice",

"address":"alice:9100",

"listen\_addr":"0.0.0.0:9100"

},

{

"party":"bob",

"address":"bob:9200",

"listen\_addr":"0.0.0.0:9200"

}

],

"runtime\_config":{

"protocol":spu.spu\_pb2.SEMI2K,

"field":spu.spu\_pb2.FM128,

"sigmoid\_mode":spu.spu\_pb2.RuntimeConfig.SIGMOID\_REAL

}

}

new\_spu=sf.SPU(cluster\_def=cluster\_def)

### ab一起

# from pathlib import Path

# import pandas as pd

import os

current\_dir=os.getcwd()

new\_spu.psi(

keys={"alice":["id"], "bob":["id"]},

input\_path={"alice":f"{current\_dir}/file\_2.csv","bob":f"{current\_dir}/file\_1.csv"},

output\_path={"alice":f"{current\_dir}/a\_output.csv", "bob":f"{current\_dir}/b\_output.csv"},

receiver="alice"

)