

DeploymentGuide

1. Prophecis简介

Prophecis 使用 `helm` 来进行 `kubernetes` 包管理，主要安装文件位于 `install` 目录下。`install` 目录包含了三个组件 `notebook-controller`，`MinioDeployment`，`Prophecis`，主体为 `Prophecis`。使用前，需要初始化MySQL数据库，并挂载NFS目录来存储数据。

2. 环境准备

2.1 机器规划

2.2 软件

软件	版本	位置
Helm	3.2.1	https://github.com/helm/helm/releases
Kubenertes	1.18.6	https://github.com/kubernetes/kubernetes
Istio	1.8.2	https://github.com/istio/istio
Docker	19.03.9	https://www.docker.com/
nfs-utils	1.3.0	
seldon-core	1.13.0	https://github.com/SeldonIO/seldon-core

- 验证Helm

```
1 $ helm version
2 version.BuildInfo{Version:"v3.2.1",
  GitCommit:"fe51cd1e31e6a202cba7dead9552a6d418ded79a",
  GitTreeState:"clean", GoVersion:"go1.13.10"}
```

- 验证Kubernertes

```
1 $ kubectl version
2 Client Version: version.Info{Major:"1", Minor:"19", GitVersion:"v1.19.3",
  GitCommit:"1e11e4a2108024935ecfcb2912226cedaafd99df",
  GitTreeState:"clean", BuildDate:"2020-10-14T12:50:19Z",
  GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
3 Server Version: version.Info{Major:"1", Minor:"18", GitVersion:"v1.18.6",
  GitCommit:"dff82dc0de47299ab66c83c626e08b245ab19037",
  GitTreeState:"clean", BuildDate:"2020-07-15T16:51:04Z",
  GoVersion:"go1.13.9", Compiler:"gc", Platform:"linux/amd64"}
```

- 验证Docker

```
1 $ docker version
```

```

2 ...
3 Client: Docker Engine - Community
4 Version:          19.03.9
5 ...
6 Server: Docker Engine - Community
7 Engine:
8 Version:          19.03.9

```

- Seldon Core相关:

```

1 #部署
2 kubectl create namespace seldon-system
3 helm install seldon-core seldon-core-operator . \ --set
  usageMetrics.enabled=true \ --namespace seldon-system \ --set
  istio.enabled=true
4 #验证
5 kubectl -n seldon-system get pods

```

- Istio相关:

```

1 #部署
2 istioctl install
3 #验证, 查看相关Pod是否正常Running
4 kubectl -n istio-system get pods

```

2.3 目录挂载(若不需共享目录可先跳过)

- NFS服务节点操作

```

1 # NFS服务节点IP地址  NFS_SERVER_IP='172.22.1.122'
2 # NFS挂载目录  NFS_PATH_LOG='/mlss/di/jobs/prophecis'
3 #
4 # 需要挂载的目录
5 # /data/bdap-ss/mlss-data/tmp
6 # /mlss/di/jobs/prophecis
7 # /cosdata/mlss-test
8 mkdir -p ${NFS_PATH_LOG}
9 # 追加写入到文件中, 标明挂载信息
10 echo "${NFS_PATH_LOG} ${NFS_SERVER_IP}/24(rw, sync, no_root_squash)">>
  /etc/exports
11 # 刷新nfs, 让服务节点使用该nfs挂载
12 exportfs -arv

```

- NFS客户端节点操作

```

1 # 需要对除Master节点外的其他节点执行挂载
2 mkdir -p ${NFS_PATH_LOG}
3 # 挂载目录
4 mount ${NFS_SERVER_IP}:${NFS_PATH_LOG} ${NFS_PATH_LOG}

```

3. 物料准备

- 安装包位置：`https://github.com/WeBankFinTech/Prophecis.git`

- 安装包目录



- 文件清单：
 - Helm Chart：`./install` 目录下 `notebook-controller` , `MinioDeployment` , `Prophecis`
 - SQL Script：`./cc/sql` 目录下 `prophecis.sql` 和 `prophecis-data.sql` 文件
- 镜像列表（安装时自动下载）：

```
1 # 版本号 VERSION=v0.2.0
2 # Docker仓库位置
3 wedatasphere/prophecis:
4 # Docker标签
5 ui-${VERSION}
6 trainer-${VERSION}
7 restapiI-${VERSION}
8 lcm-${VERSION}
9 storage-${VERSION}
10 cc-apiserverC-${VERSION}
11 cc-apigateway-${VERSION}
12 jobmonitor-${VERSION}
13 mllabis-v0.1.1
14 master-97
15
16 wedatasphere/mllabis:
17 Prophecis_1.8.0_tensorflow-2.0.0-jupyterlab-gpu-v0.5.0
```

4. 配置文件修改

需要修改 `./install/prophecis/values.yaml` 中的信息。

4.1 配置数据库访问的信息

```
1 # MySQLIP地址 DATABASE_IP='172.22.1.128'
2 # MySQL端口号 DATABASE_PORT='3306'
3 # MySQL数据库名 DATABASE_DB='mlss_db'
4 # MySQL用户名 DATABASE_USERNAME='mlss'
5 # MySQL用户密码 DATABASE_PASSWORD='123'
6 database:
7     server: ${DATABASE_IP}
8     port: ${DATABASE_PORT}
9     name: ${DATABASE_DB}
10    user: ${DATABASE_USERNAME}
11    pwd: ${DATABASE_PASSWORD}
```

4.2 配置UI的URL访问路径

```
1 # 网页访问地址 SERVER_IP='172.22.1.68'
2 # 网页访问端口 SERVER_PORT='30803'
3 server_ui_gateway: ${SERVER_IP}:30778
```

```
4 ui:
5     service:
6         bdap:
7             nodePost: ${SERVER_PORT}
```

4.3 配置LDAP

Prophecis使用LDAP来负责统一认证

```
1 # LDAP的服务地址 LDAP_ADDRESS='ldap://172.22.1.122:1389/'
2 # LDAP的DNS解析地址 LDAP_BASE_DN='dc=webank,dc=com'
3 cc:
4     ldap:
5         address: ${LDAP_ADDRESS}
6         baseDN: ${LDAP_BASE_DN}
```

5. 操作序列

5.1 数据库更新

需要在MySQL命令行内载入文件

在数据库内执行 `./cc/sql` 下的 `SQL` 文件 `prophecis.sql` 和 `prophecis-data.sql`，需要使用SQL脚本来创建表结构和初始数据

```
1 source prophecis.sql
2 source prophecis-data.sql
```

5.2 创建 namespace

Prophecis默认使用 `kubernetes` 的命名空间 `prophecis`，需要创建

```
1 kubectl create namespace prophecis
```

5.3 给运行节点标签

Prophecis使用 `kubernetes` 的节点标签来识别用途

```
1 # 服务节点的标签 LABEL_CPU='mlsskf010001 mlsskf010002'
2 # GPU节点的标签 LABEL_GPU='mlsskf010003 mlsskf010004'
3
4 kubectl label nodes ${LABEL_CPU} mlss-node-role=platform
5 kubectl label nodes ${LABEL_GPU} hardware-type=NVIDIAGPU
```

5.4 安装组件

Prophecis部署需要三个组件 `notebook-controller`，`MinioDeployment`，`Prophecis`。
部署执行目录为 `./install` 目录下。

```
1 # 安装Notebook Controller组件
2 helm install notebook-controller ./notebook-controller
3 # 安装MinIO组件
4 helm install minio-prophecis --namespace prophecis ./MinioDeployment
5 # 安装prophecis组件
6 helm install prophecis ./prophecis
```

5.5 MLFlow实验 workflow 相关(mlflow appconn安装)

- 服务配置：在values.yaml中配置Linkis相关地址
- MLFlow Appconn：将appconn相关lib部署于DSS Appconn目录下
- 数据库更新

```
1 #sql目录
2 source mlflow-sql.sql
```

- 编译及使用可参考Deployment_Documents中的Prophecis Appconn安装文档

5.6 DataSphereStudio相关

- Appconn安装：将appconn相关lib部署于DSS Appconn目录下
- 数据库更新

```
1 #sql目录
2 source mlss-sql.sql
```

- 编译及使用可参考Deployment_Documents中的Prophecis Appconn安装文档

6. 环境验证

6.1 服务验证

- `kubectl get -n prophecis pod` 查看所有服务是否正常 `Running`，若异常通过 `kubectl describe` 及 `kubectl log` 查看异常原因

6.2 登录验证

- 所有Pod正常Running后，访问 `http://${SERVER_IP}:${SERVER_PORT}`，默认账号为 `admin`，密码 `admin`。