

1.安装MetalB

[安装手册](#)

2.安装IngressController

[参考文档5.2](#)

3.配置kubefate

[参考文档6](#)

- 安装kubefate服务

```
1 cd KubeFATE-master/k8s-deploy
2 make install
```

- chart包管理

当应用安装出问题时，可以自定义修改chart包然后上传

```
1 #查看chart包
2 kubefate chart ls
3
4 #打包本地chart包
5 cd KubeFATE-master/helm-charts
6 make release
7
8 #上传本地chart包
9 kubefate chart upload -f ../helm-charts/fate-v1.6.0-a.tgz
10 kubefate chart upload -f ../helm-charts/fate-serving-v2.0.4.tgz
11 kubefate chart upload -f ../helm-charts/fate-exchange-v1.6.0-a.tgz
```

4.配置StorageClass

4.1 NFS

```
1 #rbac.yaml
2 [centos@cpu0 ~]$ cat rbac.yaml
3 kind: ClusterRole
4 apiVersion: rbac.authorization.k8s.io/v1
5 metadata:
6   name: nfs-provisioner-runner
7 rules:
8   - apiGroups: [""]
9     resources: ["persistentvolumes"]
10    verbs: ["get", "list", "watch", "create", "delete"]
11   - apiGroups: [""]
12     resources: ["persistentvolumeclaims"]
13     verbs: ["get", "list", "watch", "update"]
14   - apiGroups: ["storage.k8s.io"]
```

```

15     resources: ["storageclasses"]
16     verbs: ["get", "list", "watch"]
17 - apiGroups: [""]
18     resources: ["events"]
19     verbs: ["create", "update", "patch"]
20 - apiGroups: [""]
21     resources: ["services", "endpoints"]
22     verbs: ["get"]
23 - apiGroups: ["extensions"]
24     resources: ["podsecuritypolicies"]
25     resourceNames: ["nfs-provisioner"]
26     verbs: ["use"]
27 ---
28 kind: ClusterRoleBinding
29 apiVersion: rbac.authorization.k8s.io/v1
30 metadata:
31   name: run-nfs-provisioner
32 subjects:
33   - kind: ServiceAccount
34     name: nfs-provisioner
35     # replace with namespace where provisioner is deployed
36     namespace: default
37 roleRef:
38   kind: ClusterRole
39   name: nfs-provisioner-runner
40   apiGroup: rbac.authorization.k8s.io
41 ---
42 kind: Role
43 apiVersion: rbac.authorization.k8s.io/v1
44 metadata:
45   name: leader-locking-nfs-provisioner
46 rules:
47   - apiGroups: [""]
48     resources: ["endpoints"]
49     verbs: ["get", "list", "watch", "create", "update", "patch"]
50 ---
51 kind: RoleBinding
52 apiVersion: rbac.authorization.k8s.io/v1
53 metadata:
54   name: leader-locking-nfs-provisioner
55 subjects:
56   - kind: ServiceAccount
57     name: nfs-provisioner
58     # replace with namespace where provisioner is deployed
59     namespace: default
60 roleRef:
61   kind: Role
62   name: leader-locking-nfs-provisioner
63   apiGroup: rbac.authorization.k8s.io
64
65 #deployment.yaml
66 [centos@cpu0 ~]$ cat deployment.yaml
67 kind: StorageClass
68 apiVersion: storage.k8s.io/v1
69 metadata:
70   name: nfs
71 provisioner: example.com/nfs
72 mountOptions:

```

```
73   - vers=4.1
74   ---
75   apiVersion: v1
76   kind: ServiceAccount
77   metadata:
78     name: nfs-provisioner
79   ---
80   kind: Service
81   apiVersion: v1
82   metadata:
83     name: nfs-provisioner
84     labels:
85       app: nfs-provisioner
86   spec:
87     ports:
88       - name: nfs
89         port: 2049
90       - name: nfs-udp
91         port: 2049
92         protocol: UDP
93       - name: nlockmgr
94         port: 32803
95       - name: nlockmgr-udp
96         port: 32803
97         protocol: UDP
98       - name: mountd
99         port: 20048
100      - name: mountd-udp
101        port: 20048
102        protocol: UDP
103      - name: rquotad
104        port: 875
105      - name: rquotad-udp
106        port: 875
107        protocol: UDP
108      - name: rpcbind
109        port: 111
110      - name: rpcbind-udp
111        port: 111
112        protocol: UDP
113      - name: statd
114        port: 662
115      - name: statd-udp
116        port: 662
117        protocol: UDP
118    selector:
119      app: nfs-provisioner
120    ---
121   kind: Deployment
122   apiVersion: apps/v1
123   metadata:
124     name: nfs-provisioner
125   spec:
126     selector:
127       matchLabels:
128         app: nfs-provisioner
129     replicas: 1
130     strategy:
```

```
131     type: Recreate
132   template:
133     metadata:
134       labels:
135         app: nfs-provisioner
136     spec:
137       serviceAccount: nfs-provisioner
138       nodeSelector:
139         kubernetes.io/hostname: cpu1
140       containers:
141         - name: nfs-provisioner
142           image: k8s.gcr.io/sig-storage/nfs-provisioner:v3.0.0
143           ports:
144             - name: nfs
145               containerPort: 2049
146             - name: nfs-udp
147               containerPort: 2049
148               protocol: UDP
149             - name: nlockmgr
150               containerPort: 32803
151             - name: nlockmgr-udp
152               containerPort: 32803
153               protocol: UDP
154             - name: mountd
155               containerPort: 20048
156             - name: mountd-udp
157               containerPort: 20048
158               protocol: UDP
159             - name: rquotad
160               containerPort: 875
161             - name: rquotad-udp
162               containerPort: 875
163               protocol: UDP
164             - name: rpcbind
165               containerPort: 111
166             - name: rpcbind-udp
167               containerPort: 111
168               protocol: UDP
169             - name: statd
170               containerPort: 662
171             - name: statd-udp
172               containerPort: 662
173               protocol: UDP
174           securityContext:
175             capabilities:
176               add:
177                 - DAC_READ_SEARCH
178                 - SYS_RESOURCE
179           args:
180             - "-provisioner=example.com/nfs"
181           env:
182             - name: POD_IP
183               valueFrom:
184                 fieldRef:
185                   fieldPath: status.podIP
186             - name: SERVICE_NAME
187               value: nfs-provisioner
188             - name: POD_NAMESPACE
```

```

189         valueFrom:
190             fieldRef:
191                 fieldPath: metadata.namespace
192         imagePullPolicy: "IfNotPresent"
193         volumeMounts:
194             - name: export-volume
195               mountPath: /export
196     volumes:
197         - name: export-volume
198           hostPath:
199             path: /srv/k8s

```

4.2 Longhorn

```
1 |
```

5.安装serving

- 创建对应的namespace

```

1 kubectl create namespace fate-10000
2 kubectl create namespace fate-10000
3 kubectl create ns fate-serving-9999
4 kubectl create ns fate-serving-10000

```

- 自定义yaml文件内容

```
cluster-serving-10000.yaml
```

```

1 name: fate-serving-10000
2 namespace: fate-serving-10000
3 chartName: fate-serving
4 chartVersion: v2.0.4
5 partyId: 10000
6 registry: ""
7 imageTag: ""
8 pullPolicy:
9 persistence: true
10 istio:
11     enabled: false
12 podSecurityPolicy:
13     enabled: false
14 modules:
15     - servingProxy
16     - servingRedis
17     - servingServer
18     - servingZookeeper
19     - servingAdmin
20
21 servingAdmin:
22     ingressHost: 10000.serving-admin.kubefate.net
23     username: admin
24     password: admin
25 servingProxy:
26     type: ClusterIP

```

```
27   ingerssHost: 10000.serving-proxy.kubefate.net
28   partyList:
29     - partyId: 9999
30       partyIp: serving-proxy.fate-serving-9999
31       partyPort: 8869
32   servingServer:
33     storageClass: "nfs"
34     accessMode: ReadWriteOnce
35     size: 1Gi
36   servingRedis:
37     password: fate_dev
38     storageClass: "nfs"
39     accessMode: ReadWriteOnce
40     size: 1Gi
41   servingZookeeper:
42     storageClass: "nfs"
43     accessMode: ReadWriteOnce
44     size: 1Gi
```

cluster-serving-9999.yaml

```
1  name: fate-serving-9999
2  namespace: fate-serving-9999
3  chartName: fate-serving
4  chartVersion: v2.0.4
5  partyId: 9999
6  registry: ""
7  imageTag: ""
8  pullPolicy:
9  persistence: true
10 istio:
11   enabled: false
12 podSecurityPolicy:
13   enabled: false
14 modules:
15   - servingProxy
16   - servingRedis
17   - servingServer
18   - servingZookeeper
19   - servingAdmin
20
21 servingAdmin:
22   ingressHost: 9999.serving-admin.kubefate.net
23   username: admin
24   password: admin
25 servingProxy:
26   type: ClusterIP
27   ingerssHost: 9999.serving-proxy.kubefate.net
28   partyList:
29     - partyId: 10000
30       partyIp: serving-proxy.fate-serving-10000
31       partyPort: 8869
32   servingServer:
33     storageClass: "nfs"
34     accessMode: ReadWriteOnce
35     size: 1Gi
36   servingRedis:
```

```
37 password: fate_dev
38 storageClass: "nfs"
39 accessMode: ReadWriteOnce
40 size: 1Gi
41 servingZookeeper:
42   storageClass: "nfs"
43   accessMode: ReadWriteOnce
44   size: 1Gi
```

- 安装服务

```
1 kubefate cluster install -f ./cluster-serving-10000.yaml
2 kubefate cluster install -f ./cluster-serving-9999.yaml
3 kubefate cluster list
4 kubefate job list
5 kubefate describe job ID
```

6.安装fate集群

cluster-10000.yaml

```
1 name: fate-10000
2 namespace: fate-10000
3 chartName: fate
4 chartVersion: v1.6.0-a
5 partyId: 10000
6 registry: "hub.c.163.com/federatedai"
7 imageTag: ""
8 pullPolicy:
9 imagePullSecrets:
10 - name: myregistrykey
11 persistence: true
12 istio:
13   enabled: false
14 podSecurityPolicy:
15   enabled: false
16 modules:
17   - rollsite
18   - clustermanager
19   - nodemanager
20   - mysql
21   - python
22   - fateboard
23 backend: eggroll
24 rollsite:
25   exchange:
26     ip: rollsite
27     port: 9370
28     partyList:
29     - partyId: 9999
30       partyIp: rollsite.fate-9999
31       partyPort: 9370
32 nodemanager:
33   count: 2
34   sessionProcessorsPerNode: 4
35   storageClass: "nfs"
```

```
36     accessMode: ReadWriteOnce
37     size: 10Gi
38 python:
39     type: clusterIP
40 mysql:
41     ip: mysql
42     port: 3306
43     user: fate
44     password: fate_dev
45     storageClass: "nfs"
46     accessMode: ReadWriteOnce
47     size: 1Gi
48 serving:
49     useRegistry: true
50     zookeeper:
51         hosts:
52         - serving-zookeeper.fate-serving-10000:2181
53     use_acl: false
```

cat cluster-9999.yaml

```
1  name: fate-9999
2  namespace: fate-9999
3  chartName: fate
4  chartVersion: v1.6.0-a
5  partyId: 9999
6  registry: "hub.c.163.com/federatedai"
7  imageTag: ""
8  pullPolicy:
9  persistence: true
10 istio:
11     enabled: false
12 podSecurityPolicy:
13     enabled: false
14 modules:
15     - rollsite
16     - clustermanager
17     - nodemanager
18     - mysql
19     - python
20     - fateboard
21 backend: eggroll
22
23 rollsite:
24     exchange:
25         ip: rollsite
26         port: 9370
27         partyList:
28         - partyId: 10000
29           partyIp: rollsite.fate-10000
30           partyPort: 9370
31 nodemanager:
32     count: 2
33     sessionProcessorsPerNode: 4
34     storageClass: "nfs"
35     accessMode: ReadWriteOnce
36     size: 20Gi
```



```

37 python:
38   type: ClusterIP
39 mysql:
40   ip: mysql
41   port: 3306
42   user: fate
43   password: fate_dev
44   storageClass: "nfs"
45   accessMode: ReadWriteOnce
46   size: 1Gi
47 serving:
48   useRegistry: true
49   zookeeper:
50     hosts:
51     - serving-zookeeper.fate-serving-9999:2181
52   use_acl: false

```

- 安装fate集群

```

1 kubefate cluster install -f ./cluster-10000.yaml
2 kubefate cluster install -f ./cluster-9999.yaml
3 kubefate cluster install -f cluster-serving-10000.yaml
4 kubefate cluster install -f cluster-serving-9999.yaml

```

- job管理以及kubefate命令

```

1 #常用命令
2 kubefate job list
3 kubefate job describe
4 kubefate cluster list
5
6 [ppc@cpu7 k8s-deploy]$ kubefate
7 NAME:
8     kubefate - A new cli application
9
10 USAGE:
11     kubefate [global options] command [command options] [arguments...]
12
13 COMMANDS:
14     chart      List Charts, create, delete and describe a Chart
15     cluster    Manage Cluster install, delete and update
16     job        List jobs, describe and delete a job
17     namespace  List namespace
18     service    Start KubeFATE as service
19     user       List all users and describe a user's info
20     version    Show kubefate command line and service version
21     help, h    Shows a list of commands or help for one command
22
23 GLOBAL OPTIONS:
24     --help, -h  Show help (default: false)

```

7.进入容器跑任务

```
1 | kubectl exec -n fate-9999 -it svc/fateflow -c python -- bash
2 | kubectl exec -n fate-10000 -it svc/fateflow -c python -- bash
```

8.页面访问

页面访问可以讲各ingress的域名映射写到host表里之后直接访问。