链滴

Docker Swarm 部署 micro 微服务应用

作者：Allenxuxu

原文链接：https://ld246.com/article/1563453907974

来源网站：链滴

微服务应用使用容器部署非常方便，但是当应用服务注册自身地址(ip:port)到服务注册中心的时候，果注册的是容器内的ip，别的服务是无法访问到的。

解决这个问题，可以在运行容器的时候指定网络模式为 host (--net=host) ，这样就可以跳过 Docker 的独立网络栈，直接通过本机IP端口就可以访问，但是这样会大量占用本地端口。

最好的场景还是后端服务都在容器网络中，仅 API 网关暴露一个端口供外部访问，但是同时还后端服还需要能实现跨机器的网络连通。

早期 Docker 本身的容器网络本身并不支持跨机器，也就是说明如果容器部署在不同的节点（服务器上面，只能通过暴露端口到宿主机上，再通过宿主机之间进行通信。Docker 12.0 之后的版本自带 Doker Swarm，Docker Swarm 的 Overlay 网络驱动可以实现跨主机网络通信。Kubernetes 固然好，是同时也非常重，学习成本也很大，Swarm 在小项目中还是有用武之地的。

dokcer swarm 集群搭建

准备两台安装有 docker 的机器：

192.168.0.1

192.168.0.2

192.168.0.1 创建master节点

```
#    docker swarm init

#    docker swarm join \
>     --token SWMTKN-1-3uu3gjkdt6xgk06wd1c9gfog8xec99ga69ilcclyzyk181n5ki-6f7frw75g
pdwsl1yvpf885lw \
>     192.168.0.1:2377
This node joined a swarm as a worker.
```

复制上面的 docker swarm join ...

在 192.168.0.2 上执行，即将本机加入 swarm 集群。

至此，我们已经创建了一个最基础的 swarm 的集群，执行命令查看：

```
#   docker node ls
ID                           HOSTNAME   STATUS  AVAILABILITY  MANAGER STATUS
r76ighlnw0p2r0tbd9wmoqaep    server2    Ready   Active
rzqbzl58hlu89xoty4cedn0er *  server1    Ready   Active        Leader
```

创建 overlay 网络

先创建一个可以跨机器的 overlay 网络

```
docker network create -d overlay my_net
```

部署应用服务

部署服务注册中心 consul

在服务器 192.168.0.1 中使用 Docker 简单部署一个使用。

```
docker run --name consul -d -p 8500:8500/tcp consul agent -server -ui -bootstrap-expect=1
client=0.0.0.0
```

部署 API 网关

采用 micro 官方的 micro api，不了解 micro 的可以看我之前的博客，或者去 micro 官方仓库查看。

```
docker service create --replicas 4  --publish published=8898,target=8080  --name micro-p -e
MICRO_REGISTRY=consul  -e MICRO_REGISTRY_ADDRESS=192.168.0.1:8500 -e  MICRO_API_
ANDLER=http  --network=my_net microhq/micro:latest api
```

部署后端服务

编写一个简单的 micro web 服务

```
package main

import (
    "log"

    "github.com/gin-gonic/gin"
    "github.com/micro/go-micro/web"
)

type Say struct{}


func (s *Say) Anything(c *gin.Context) {
    log.Print("Received Say.Anything API request")
    c.JSON(200, map[string]string{
```

```
        "message": "Hi, this is the Greeter API",
    })
}

func main() {
    // Create service
    service := web.NewService(
        web.Name("go.micro.api.greeter"),
    )

    service.Init()

    // Create RESTful handler (using Gin)
    say := new(Say)
    router := gin.Default()
    router.GET("/greeter", say.Anything)

    // Register Handler
    service.Handle("/", router)

    // Run server
    if err := service.Run(); err != nil {
        log.Fatal(err)
    }
}
```

Dockerfile 如下：

```
FROM alpine:latest
RUN apk --no-cache add ca-certificates
COPY hello-gin /hello-gin
ENTRYPOINT /hello-gin

LABEL Name=hello-gin Version=0.0.1
```

将 Docker build 出来推到自己的 Docker 仓库上，或者直接 pull 我的镜像。

部署服务

```
 docker service create --replicas 2 --name hello-xx -e MICRO_REGISTRY=consul  -e MICRO_R
GISTRY_ADDRESS=192.168.0.1:8500 --network=xuxu_net  xuxu123/hello-gin:v0.1.0
```

测试demo

```
curl --request GET  --url http://192.168.0.1:8080/greeter
```

总结

主要简单演练了一遍Docker Swarm 集群部署以及微服务部署的一个简单场景部署。相较于 K8S 的大功能，Swarm 似乎显得有些多余，但是 Swarm 的简单明了在小厂中未必有没有用武之地吧。