Swarm 系列 5-- 手动配置 Swarm

作者：james
原文链接：https://ld246.com/article/1520336560725
来源网站：链滴

手动配置swarm

- swarm项目参考地址： https://github.com/docker/swarm
- 可以下载编译swarm来手动运行swarm。swarm是一个容器编排工具，go编译出一个可执行文件配合相应的配置即可运行。

1. 手动运行

- 编译后手动运行，参考： Docker Swarm discovery
- 需要自定义配置服务发现服务，代码中默认是使用docker Hub的开放注册服务。生产环境需要配自己的distributed key/value store注册服务。
- 所以本地采用etcd

```
启动etcd
james@james-CW65:/media/james/linux/yangz/software/etcd/etcd-v3.3.0-rc.1-linux-amd64 >
./etcd --config-file etcd_default.conf
2018-03-06 11:51:20.890647 I | etcdmain: Loading server configuration from "etcd_default.con
"
2018-03-06 11:51:20.893130 I | etcdmain: etcd Version: 3.3.0-rc.1
2018-03-06 11:51:20.893167 I | etcdmain: Git SHA: d3c2acf
2018-03-06 11:51:20.893185 I | etcdmain: Go Version: go1.9.2
2018-03-06 11:51:20.893202 I | etcdmain: Go OS/Arch: linux/amd64
2018-03-06 11:51:20.893219 I | etcdmain: setting maximum number of CPUs to 4, total numb
r of available CPUs is 4
2018-03-06 11:51:20.893242 W | etcdmain: no data-dir provided, using default data-dir ./defa
lt.etcd
2018-03-06 11:51:20.893342 N | etcdmain: the server is already initialized as member before,
tarting as etcd member...
2018-03-06 11:51:20.893452 I | embed: listening for peers on http://172.19.128.93:2380
2018-03-06 11:51:20.893481 I | embed: pprof is enabled under /debug/pprof
2018-03-06 11:51:20.893526 I | embed: listening for client requests on 172.19.128.93:2379

加入集群，在每个机器运行swarm agent。可以看到默认每分钟，agent去注册一次。
--advertise参数是Address of the Docker Engine joining the cluster. Swarm manager(s) MUST
e able to reach the Docker Engine at this address. [$SWARM_ADVERTISE]
代码中的限制是：--advertise should be of the form ip:port or hostname:port
所以使用swarm必须通过tcp方式与docker daemon通信，不能用sock文件了。
MacBook-Air:src air$ swarm join --advertise=172.19.128.148:2375 etcd://172.19.128.93:2379
INFO[0000] Initializing discovery without TLS
INFO[0000] Registering on the discovery service every 1m0s...  addr=172.19.128.148:2375 dis
overy=etcd://172.19.128.93:2379
INFO[0060] Registering on the discovery service every 1m0s...  addr=172.19.128.148:2375 dis
overy=etcd://172.19.128.93:2379
INFO[0120] Registering on the discovery service every 1m0s...  addr=172.19.128.148:2

开启swarm管理节点：
--host, -H [--host option --host option]            ip/socket to listen on [$SWARM_HOST]

MacBook-Air:ac-agent air$ swarm manage -H tcp://172.19.128.148:2377  etcd://172.19.128.93
2379
INFO[0000] Initializing discovery without TLS
```

```
INFO[0000] Listening for HTTP                    addr=172.19.128.148:2377 proto=tcp
```

开始使用swarm:
可以看出在该节点上使用swarm的管理节点来获取集群信息，由于swarm是安装在一个没装docker机器上，节点也是。所以没有获取到节点的容器任何信息。

```
james@james-CW65:/media/james/linux/yangz/software/etcd/etcd-v3.3.0-rc.1-linux-amd64 >
docker -H tcp://172.19.128.148:2377  info
Containers: 0
 Running: 0
 Paused: 0
 Stopped: 0
Images: 0
Server Version: swarm/1.2.8
Role: primary
Strategy: spread
Filters: health, port, containerslots, dependency, affinity, constraint, whitelist
Nodes: 1
 (unknown): 172.19.128.148:2375
  └ ID:
  └ Status: Pending
  └ Containers: 0
  └ Reserved CPUs: 0 / 0
  └ Reserved Memory: 0 B / 0 B
  └ Labels:
  └ Error: Cannot connect to the Docker daemon at tcp://172.19.128.148:2375. Is the docker d
emon running?
  └ UpdatedAt: 2018-03-06T04:11:37Z
  └ ServerVersion:
Plugins:
 Volume:
 Network:
 Log:
Swarm:
 NodeID:
 Is Manager: false
 Node Address:
Kernel Version: 16.7.0
Operating System: darwin
Architecture: amd64
CPUs: 0
Total Memory: 0B
Name: MacBook-Air.local
Docker Root Dir:
Debug Mode (client): false
Debug Mode (server): false
Experimental: false
Live Restore Enabled: false

WARNING: No kernel memory limit support
```

```
james@james-CW65:/media/james/linux/yangz/software/etcd/etcd-v3.3.0-rc.1-linux-amd64 >
./etcdctl --endpoint http://172.19.128.93:2379 get /docker/swarm/nodes/172.19.128.148:2375
172.19.128.148:2375
```

● 在另一台机器上使用swarm

```
加入集群，指定注册地址和服务发现服务地址：
james@james-CW65:/media/james/linux/yangz/software/docker > ./swarm_linux join --advert
se=172.19.128.93:2375 etcd://172.19.128.93:2379
INFO[0000] Initializing discovery without TLS
INFO[0000] Registering on the discovery service every 1m0s...  addr=172.19.128.93:2375 disc
very=etcd://172.19.128.93:2379
会显示2个节点，但是由于docker没有开放tcp连接，没办法获取docker信息
james@james-CW65:/media/james/linux/yangz/software/etcd/etcd-v3.3.0-rc.1-linux-amd64 >
docker -H tcp://172.19.128.148:2377 info
Containers: 0
 Running: 0
 Paused: 0
 Stopped: 0
Images: 0
Server Version: swarm/1.2.8
Role: primary
Strategy: spread
Filters: health, port, containerslots, dependency, affinity, constraint, whitelist
Nodes: 2
 (unknown): 172.19.128.148:2375
  └ ID:
  └ Status: Pending
  └ Containers: 0
  └ Reserved CPUs: 0 / 0
  └ Reserved Memory: 0 B / 0 B
  └ Labels:
  └ Error: Cannot connect to the Docker daemon at tcp://172.19.128.148:2375. Is the docker d
emon running?
  └ UpdatedAt: 2018-03-06T05:18:00Z
  └ ServerVersion:
 (unknown): 172.19.128.93:2375
  └ ID:
  └ Status: Pending
  └ Containers: 0
  └ Reserved CPUs: 0 / 0
  └ Reserved Memory: 0 B / 0 B
  └ Labels:
  └ Error: Cannot connect to the Docker daemon at tcp://172.19.128.93:2375. Is the docker da
mon running?
  └ UpdatedAt: 2018-03-06T06:16:52Z
  └ ServerVersion:
Plugins:
 Volume:
 Network:
 Log:
Swarm:
 NodeID:
 Is Manager: false
 Node Address:
Kernel Version: 16.7.0
Operating System: darwin
Architecture: amd64
```

```
CPUs: 0
Total Memory: 0B
Name: MacBook-Air.local
Docker Root Dir:
Debug Mode (client): false
Debug Mode (server): false
Experimental: false
Live Restore Enabled: false

WARNING: No kernel memory limit support


MacBook-Air:src air$ swarm list  etcd://172.19.128.93:2379
INFO[0000] Initializing discovery without TLS
172.19.128.148:2375
172.19.128.93:2375
MacBook-Air:src air$
```

- 注意这里swarm仅仅提供一个集群功能。将docker连成了一个整体。但是在该集群上吗并不能使用版docker的命令行或API client来操作服务，会报错404。
- 不要混淆了独立swarm和swarm mode的区别。这里我们创建的是swarm集群，并不提供服务级的功能。参考：使用Docker Toolbox 创建Swarm集群的问题-概念混淆导致
- 实际运行这样也是不能创建服务的。发现只有使用docker swarm init来从docker Hub上获取的tokn才能使用swarm mode。为什么不在docekr swarm init的时候提供kv数据存储的参数，来本地注呢。看swarm源代码也可以看出再创建集群的时候，使用的是默认的docker hub的数据库，真不知这个为什么不让用户用自己的。

```
const tokenDeprecationErr = "Token based discovery is now deprecated and might be remov
d in the future.\nIt will be replaced by a default discovery backed by Docker Swarm Mode.\nO
her mechanisms such as consul and etcd will continue to work as expected.\n"

func create(c *cli.Context) {
    if len(c.Args()) != 0 {
        log.Fatalf("the `create` command takes no arguments. See '%s create --help'.", c.App.Na
e)
    }
    discovery := &token.Discovery{}
    discovery.Initialize("", 0, 0, nil)
    token, err := discovery.CreateCluster()
    if err != nil {
        log.Fatal(err)
    }
    fmt.Fprintf(os.Stderr, tokenDeprecationErr)
    fmt.Println(token)
}
```