29 分钟使用新 Docker Swarm Mode 搭建 Raspberry Pi 3 集群

Phodal Huang

October 24, 2017

目录

原文链接：https://www.wandianshenme.com/play/setup-docker-swarm-mode-in-raspberry-pi-3

Hellllooooo，大家！我知道花了一段时间，因为我写了关于这里的任何事情。过去的一周里，我一直在工作休假。终于有时间可以搞一些，我想要做的事情！今天我想告诉你，我将集群管理解决方案部署到，四处坐落在我公寓里的 Raspberry Pi 3。如果您在过去一两年中一直躲在岩石之下，开放源码集群管理已经成为技术领域的一大难题，让团队能够在分布式应用/框架之间提供有效的资源隔离和共享。在工作中，我和 Mesos（和 Marathon ）一起玩了很多项目，所以我以此为契机，首先尝试了 Kubernetes。有很多相当优秀的博客文章，详细介绍了其他人如何设法从头开始提升他们的集群。特别是，我看了以下帖子/项目：

1. https://github.com/luxas/kubernetes-on-arm
2. http:// blog.kubernetes.io/ 2015/11/ creating-a-Raspberry-Pi-cluster-running-Kubernetes-the-shopping-list-Part-1.html
3. http://blog.kubernetes.io/2015/12/creating-raspberry-pi-cluster-running.html
4. https:// medium.com/ google-cloud/ everything-you-need-to-know-about-the-kubernetes-raspberry-pi-cluster-2a2413bfa0fa#.9k4t8rusk
5. https://github.com/Project31/ansible-kubernetes-openshift-pi3/tree/hypriot

每个解决方案，都拥有很好的记录和简单的过程。但我遇到了一些问题，很可能是因为我试图使用最新的硬件/软件，而不是博客文章中提到的旧的、锁定的版本。任何玩过这些新潮技术的人都知道，一个星期内，事情可能会发生很大的变化，更不用说一年了。我想使用我的 Raspberr Pi 3（不是 Raspberry Pi 2）以及 Hypriot v0.8.0 Barbossa（这么多优秀的新功能和更新的核心包）。此外，根据社区来看，Hypriot v0.8.0 Barbossa 在 Raspberry Pi 3 上运行，似乎是一个令人难以置信的强大的组合。我不想错过以上的任何一个，特别是当我只有一个星期来做这些事情之前，再次淹没在工作中。过去的某些时候，我使用了一下 @Quintus23M AKA Dieter Reuter，他告诉了我：可爱的人们在 Docker 与 v1.12.0 一起制作一些魔法。以下，是我使用新的 Docker Swarm Mode 设置我的群集的经验！

步骤 **0**：我的设置 **-** 硬件

如果你想按照我的确切步骤，这里是我的确切设置：

- 4 x Raspberry Pi 3 B 型
- 4 x 三星 EVO 32GB 10 class Micro SDHC 卡与适配器（注意：32GB 不是必需的，我只是想要额外的空间）

- 4 x 微型 USB 电缆
- 1 个 Manhattan （曼哈顿）7 个端口的 USB 2.0 Ultra Hub
- 1 个 NETGEAR 5 端口的千兆以太网 10/100/1000Mbps 交换机（GS205）
- 4 个以太网 Cat-5 电缆
- 1 x Sabrent Premium 3 端口铝 USB 3.0 集线器，带多合一读卡器（12" 电缆）
- 1 x GeouxRobot Raspberry Pi 3 B 型 4 层

准备步骤：准备步骤 - 具有 **Hypriot v0.8.0** 的闪存 **Micro SD** 卡

首先，我需要下载 Hypriot v0.8.0 镜像。由于我使用的是 OS X，所以我可以利用 Hypriot 团队开发的 Flash CLI 工具（https://github.com/hypriot/flash）。在 Raspberry Pi 甚至打开之前，这个小家伙非常有效地照顾镜像定制。在这里，您可以预先定义适当的主机名，WiFi 设置等。除此之外，它内置了错误处理，可以卸载 SD 卡进行写入，一旦完成就提示您。必须有工具。对于每个 SD 卡，我使用以下主机名，`scarlett-kub-master`，`scarlett-kub-slave1`，`scarlett-kub-slave2`，`scarlett-kub-slave3` 运行命令。（注意：主机名与使用 k8s 时一样）。

```
# NOTE: ignore the $ signs, they represent your bash prompt
$ flash --hostname scarlett-kub-master --config device-init.yaml
    https://github.com/hypriot/image-builder-rpi/releases/download/v0.8.0/hypriotos-rpi-v0.8.
```

准备好出发！

准备步骤：设置无密码 **SSH** 访问

当处理 w/ 多个节点和 SSH Access 时，使用 SSH 密钥连接到远程计算机是必须的。您不需要每次与您的远程服务器交互时输入密码；这是疯狂。要设置无密码认证，我做了以下操作（注意：这假设您已经创建了一个私钥；如果没有，请参阅本指南，了解如何获取设置 http://www.tecmint.com/ssh-passwordless-login-using-ssh-keygen-in-5-easy-steps/ ）：

A. 首先，您需要获取每个 Raspberry Pi 在 LAN 上运行的 IP 地址。如果您在命令行上精通，可以使用像 arp-scan 这样的东西。如果您更喜欢 GUI，请尝试使用 LanScan Pro。我使用 Hypriot 的自定义 bash 函数来对每个 Raspberry Pi 上配置的 avahi-daemon 进行查找：

```
# NOTE: Default password for the "pirate" user is "hypriot"
$ function getip() { (traceroute $1 2>&1 | head -n 1 | cut -d\( -f 2 | cut
    -d\) -f 1) }
```

```
$ export MASTER=$(getip scarlett-kub-master.local)
$ echo $MASTER$ export SLAVE_1=$(getip scarlett-kub-slave.local)
$ echo $SLAVE_1
$ export SLAVE_1=$(getip scarlett-kub-slave1.local)
$ echo $SLAVE_1
$ export SLAVE_2=$(getip scarlett-kub-slave2.local)
$ echo $SLAVE_2
$ export SLAVE_3=$(getip scarlett-kub-slave3.local)
$ echo $SLAVE_3
$ ssh-copy-id -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no
    pirate@$SLAVE_1
$ ssh-copy-id -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no
    pirate@$SLAVE_2
$ ssh-copy-id -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no
    pirate@$SLAVE_3
```

B. 现在要测试一切工作，我跑了：

```
$ ssh pirate@scarlett-kub-master.local
```

NICE!

步骤 **1**：设置 **SSH Autocomplete**

A. 打开 `~/.ssh/config` 并添加一个如下所示的条目：

```
HOST scarlett-kub-master
 HostName scarlett-kub-master.local
 Port 22
 User pirate
 IdentityFile ~/.ssh/id_rsa_ssh
 ServerAliveInterval 60
 ServerAliveCountMax 2
 ForwardX11 yes
 PasswordAuthentication no
 IdentitiesOnly yes
 LogLevel FATAL
 ForwardAgent yes
```

B. 将这行添加到 ~/.bash_profile，然后运行 source ~/.bash_profile：

```
# Add tab completion for SSH hostnames based on ~/.ssh/config, ignoring wildcards
[[ -e "$HOME/.ssh/config" ]] && complete -o "default" -o "nospace" -W "$(grep "^Host" ~/.ssh/config | grep -v "[?*]" | cut -d " " -f2 | tr ' ' '\n')" scp sftp ssh
```

C. 重新加载你的 shell，键入 scarlett-kub-[TAB]，它应该工作。对每个其他 RPi 重复。

步骤 2：在每个节点上安装 **Docker 1.12.0-rc3**

截至今天，7/8/2016 Docker 1.12.0 是一个发布候选人，rc3 是准确的。幸运的是，@Quintus23M 能够将我自定义编译的.deb 包从上游构建！以下是安装它的步骤：

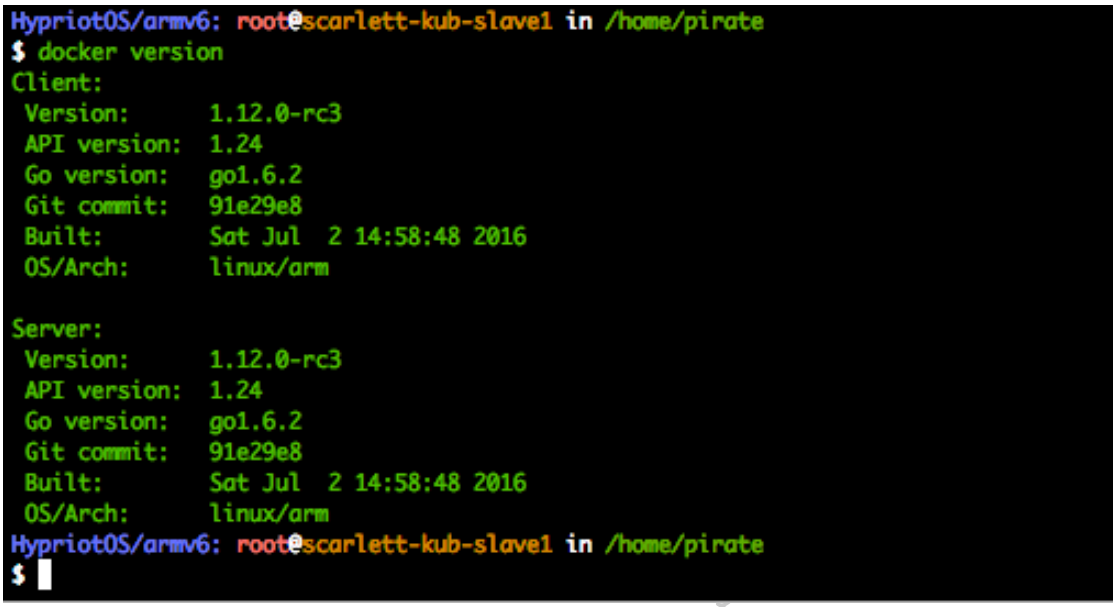
A. 抓住二进制码，然后将其扫描到每台机器上：

```
$ wget https://jenkins.hypriot.com/job/armhf-docker/15/artifact/bundles/latest/build-deb/raspbia
$ scp docker-engine_1.12.0~rc3-0~jessie_armhf.deb pirate@scarlett-kub-master.local:.
$ scp docker-engine_1.12.0~rc3-0~jessie_armhf.deb pirate@scarlett-kub-slave1.local:.
$ scp docker-engine_1.12.0~rc3-0~jessie_armhf.deb pirate@scarlett-kub-slave2.local:.
$ scp docker-engine_1.12.0~rc3-0~jessie_armhf.deb pirate@scarlett-kub-slave3.local:.
```

B. 删除当前安装的旧 Docker-hypriot，然后安装 rc3（为每台机器重复）：

```
$ ssh pirate@scarlett-kub-master.local sudo apt-get purge -y docker-hypriot
$ ssh pirate@scarlett-kub-master.local sudo dpkg -i docker-engine_1.12.0~rc3-0~jessie_armhf.deb
```

C. SSH 到机器，运行 docker 版本，你应该看到下图：



看起来不错。

步骤 3：启动 Master!

相信与否，这只需要一个命令。运行以下命令：

```
$ ssh pirate@scarlett-kub-master.local docker swarm init
Swarm initialized: current node (1njlvzi9rk2syv3xojw217o0g) is now a
    manager.
```

是的... 就是这样。通过运行以下内容（ssh 到主节点）来验证一切是否正常：

```
$ docker node ls
ID HOSTNAME MEMBERSHIP STATUS AVAILABILITY MANAGER STATUS
al55dh1yjho2wojhagzksdqwu * scarlett-kub-master Accepted Ready Active Leader
```

Amazing.

步骤 4：启动一些节点

只是当你以为它不能很容易得到东西...从 OS X 运行（注意：假设 bash 变量 $MASTER 是您的主节点的 IP 地址）：

```
$ function getip() { (traceroute $1 2>&1 | head -n 1 | cut -d\( -f 2 | cut
    -d\) -f 1) }
$ export MASTER=$(getip scarlett-kub-master.local)
```

```
$ ssh pirate@scarlett-kub-slave1.local docker swarm join $MASTER:2377
$ ssh pirate@scarlett-kub-slave2.local docker swarm join $MASTER:2377
$ ssh pirate@scarlett-kub-slave3.local docker swarm join $MASTER:2377
```

BOOM。再次运行该验证命令，您应该看到如下：

```
$ docker swarm init
Swarm initialized: current node (al55dh1yjho2wojhagzksdqwu) is now a manager.
HypriotOS/armv7: root@scarlett-kub-master in /home/pirate
$ history
    1  cd ~pirate/
    2  ls
    3  sudo apt-get purge -y docker-hypriot && sudo dpkg -i docker-engine_1.12.0~rc3-0~jessie_armhf.deb
    4  docker swarm init
    5  history
HypriotOS/armv7: root@scarlett-kub-master in /home/pirate
$ docker node ls
ID                           HOSTNAME            MEMBERSHIP  STATUS  AVAILABILITY  MANAGER STATUS
6ex4mfoeetyc4kgi4a0q0mdp8    scarlett-kub-slave2  Accepted    Ready   Active
al55dh1yjho2wojhagzksdqwu *  scarlett-kub-master  Accepted    Ready   Active        Leader
b9tf2tqr6cabcew7i4z257bhz    scarlett-kub-slave1  Accepted    Ready   Active
c5j8z8ora8bjq0meu997vn1ut    scarlett-kub-slave3  Accepted    Ready   Active
HypriotOS/armv7: root@scarlett-kub-master in /home/pirate
$
```

不能相信吗？那我们还没有完成！是时候启动一些容器。

步骤 **5**：让我们创建一个简单的 **ping Google DNS** 服务器的容器：

```
# run this from the master
$ docker service create --name ping hypriot/rpi-alpine-scratch ping 8.8.8.8
0ktotb58pbi0xp9op6anrdmkp
# verify the task was created
$ docker service tasks ping
ID NAME SERVICE IMAGE LAST STATE DESIRED STATE NODE
706fyhabjxfxgtcwesobkkokz ping.1 ping hypriot/rpi-alpine-scratch Running 14
    seconds Running scarlett-kub-master
Hypriot/armv7: pirate@scarlett-kub-master in ~
$
```

WOOT!

步骤 **6**：放大任务数量！来 **10** 个坏男孩怎么样？没问题，让我们来运行一下：

```
# update to 10
$ docker service update ping --replicas 10
ping
```

```
Hypriot/armv7: pirate@scarlett-kub-master in ~
$
# verify$ docker service tasks ping
ID NAME SERVICE IMAGE LAST STATE DESIRED STATE NODE
706fyhabjxfxgtcwesobkkokz ping.1 ping hypriot/rpi-alpine-scratch Running
    About a minute Running scarlett-kub-master
9lsto7r7qt1n2bua9ltirgll4 ping.2 ping hypriot/rpi-alpine-scratch Running 14
    seconds Running scarlett-kub-slave1
4t0mpw823usq0jbm1lymh5ga2 ping.3 ping hypriot/rpi-alpine-scratch Running 14
    seconds Running scarlett-kub-slave2
0vc51yzpe29qfzrz9uk3ffkx3 ping.4 ping hypriot/rpi-alpine-scratch Running 14
    seconds Running scarlett-kub-master
2s01xvf4sjb7k558a7ga3b7hs ping.5 ping hypriot/rpi-alpine-scratch Running 14
    seconds Running scarlett-kub-slave3
by7osvlr4wb7474brrlouv9ad ping.6 ping hypriot/rpi-alpine-scratch Running 14
    seconds Running scarlett-kub-slave3
17uxtvgqa29hul6kuqs8penv3 ping.7 ping hypriot/rpi-alpine-scratch Running 14
    seconds Running scarlett-kub-slave2
b6q27d3hmohr70f2odjuycyln ping.8 ping hypriot/rpi-alpine-scratch Running 14
    seconds Running scarlett-kub-slave1
7rri99dki0fyj012cbenzi3o1 ping.9 ping hypriot/rpi-alpine-scratch Running 14
    seconds Running scarlett-kub-slave2
1zmgn86bqtxj68ohxxo5226id ping.10 ping hypriot/rpi-alpine-scratch Running
    14 seconds Running scarlett-kub-slave1
Hypriot/armv7: pirate@scarlett-kub-master in ~
$
# Alternative approach to scaling:
$ docker service scale ping=10
ping scaled to 10
```

而且，我们完成了：

```
HypriotOS/armv7: root@scarlett-kub-master in /home/pirate
$ docker service tasks ping
ID                         NAME     SERVICE  IMAGE                       LAST STATE        DESIRED STATE  NODE
3rpejo9r3a91nw6v1wu04ig4h  ping.1   ping     hypriot/rpi-alpine-scratch  Running 14 hours  Running        scarlett-kub-master
c7m082244o3s617ix0f5un6qx  ping.2   ping     hypriot/rpi-alpine-scratch  Running 14 hours  Running        scarlett-kub-slave3
23p97rl543ylr5e6q203h2oa7  ping.3   ping     hypriot/rpi-alpine-scratch  Running 14 hours  Running        scarlett-kub-master
5zxy8x0uec2nfe3dmphupofvz  ping.4   ping     hypriot/rpi-alpine-scratch  Running 14 hours  Running        scarlett-kub-slave2
6jgbx8wf4vheqe41fs99y7h3t  ping.5   ping     hypriot/rpi-alpine-scratch  Running 14 hours  Running        scarlett-kub-slave2
d7dd4qyl5zsruxxxj5nf3rlg9  ping.6   ping     hypriot/rpi-alpine-scratch  Running 14 hours  Running        scarlett-kub-slave3
11qqhz6p7h10vzvitf57uxfpv  ping.7   ping     hypriot/rpi-alpine-scratch  Running 14 hours  Running        scarlett-kub-slave1
djqfer82440zglezwy42vzn99  ping.8   ping     hypriot/rpi-alpine-scratch  Running 14 hours  Running        scarlett-kub-slave1
62u6tur72vj60prh8eenc8342  ping.9   ping     hypriot/rpi-alpine-scratch  Running 14 hours  Running        scarlett-kub-slave3
68oyfvrz90w34mt4pzbnvxsd9  ping.10  ping     hypriot/rpi-alpine-scratch  Running 14 hours  Running        scarlett-kub-slave2
HypriotOS/armv7: root@scarlett-kub-master in /home/pirate
$
```

步骤 7：坐在后面敬畏

很简单。Bravo 到 Hypriot 和 Docker。你们刚刚做了整整一个礼拜！最疯狂的部分是，这是我第一次使用 HypriotOS 和 Docker 在一起！29 分钟即将完成。疯狂。

感谢您跟随旅程，请不要犹豫与我联系或放弃评论，如果这对您有用！

进一步阅读：

1. http:// blog.hypriot.com/ post/ swarm-machines-or-having-fun-with-docker-machine-and-the-new-docker-swarm-orchestration/
2. http://blog.hypriot.com/
3. 阅读更多关于 Docker 1.12.0 的 Swarm Mode 出现在这里：https://docs.docker.com/engine/swarm/

原文链接：How I setup a Raspberry Pi 3 Cluster Using The New Docker Swarm Mode In 29 Minutes

原文链接：https://www.wandianshenme.com/play/setup-docker-swarm-mode-in-raspberry-pi-3