StarFive
赛昉科技

在昉·星光 2上运行Docker

应用说明

版本：1.1

日期：2023/06/05

Doc ID: VisionFive2-ANCH-008

法律声明

阅读本文件前的重要法律告知。

版权注释

本文档中的说明均基于“视为正确”提供，可能包含部分错误。内容可能因产品开发而定期更新或修订。上海赛昉科技有限公司（以下简称“赛昉科技”）保留对本协议中的任何内容进行更改的权利，恕不另行通知。

赛昉科技明确否认任何形式的担保、解释和条件，无论是明示的还是默示的，包括但不限于适销性、特定用途适用性和非侵权的担保或条件。

赛昉科技无需承担因应用或使用任何产品或电路而产生的任何责任，并明确表示无需承担任何及所有连带责任，包括但不限于间接、偶然、特殊、惩戒性或由此造成的损害。

联系我们：

地址： 浦东新区盛夏路61弄张润大厦2号楼502，上海市，201203，中国

网站： http://www.starfivetech.com

邮箱：sales@starfivetech.com （销售） support@starfivetech.com（支持）

前言

关于本指南和技术支持信息

关于本手册

本应用说明手册介绍如何在昉·星光 2上运行Docker。

修订历史

表 0-1 修订历史

版本	发布说明	修订
1.0	2023/04/25	首次发布。
1.1	2023/06/05	• 在制作内核镜像 (第 8页)中增加注释及修正笔误。 • 修正制作内核镜像 (第 8页)中的笔误及第三步的生成路径。 • 修正制作内核镜像 (第 8页)中`./VisionFive2/work/boot`路径下文件名称。

注释和注意事项

本指南中可能会出现以下注释和注意事项:

- **提示:**
  建议如何在某个主题或步骤中应用信息。
- **注:**
  解释某个特例或阐释一个重要的点。
- **重要:**
  指出与某个主题或步骤有关的重要信息。
- **警告:**
  表明某个操作或步骤可能会导致数据丢失、安全问题或性能问题。
- **警告:**
  表明某个操作或步骤可能导致物理伤害或硬件损坏。

目录

表格清单 5
插图清单 6
法律声明 ii
前言 iii
1. 简介 7
2. 准备 8
2.1. 准备硬件 8
2.2. 准备内核镜像 8
2.2.1. 制作内核镜像 8
2.2.2. 在昉·星光 2上更新内核镜像 10
3. 在昉·星光 2上启动Docker 12

表格清单

表 0-1 修订历史 iii
表 2-1 硬件准备 8

插图清单

图 2-1 示例运行过程.....9
图 2-2 昉·星光 2启动输出示例.....10
图 2-3 示例输出.....10
图 3-1 示例输出.....12
图 3-2 示例输出.....12
图 3-3 示例输出.....13
图 3-4 示例命令与输出.....13
图 3-5 示例命令与输出.....14

1. 简介

Docker是一个开源的应用容器引擎，开发者可以利用它打包应用和依赖包到一个轻量级、可移植的容器中，然后发布到任何版本的Linux设备上，能够更高效的利用系统资源、保证一致的运行环境，实现持续交付和部署，以及后期更轻松的迁移、维护、扩展。

本应用说明手册介绍如何在昉·星光 2上运行Docker。

2. 准备

在运行Docker前，请确保作好以下准备。

1. 准备硬件 (第 8页)
2. 准备内核镜像 (第 8页)

2.1. 准备硬件

在执行演示程序之前，请务必准备以下硬件：

表 2-1 硬件准备

类型	M/O*	项目	注释
通用	M	昉·星光 2 单板计算机	-
通用	M	• 容量不低于32 GB的Micro-SD卡 • Micro-SD卡读卡器 • 计算机（Windows/Mac OS/Linux） • USB转串口转换器（3.3 V I/O，带线） • 以太网电缆 • 电源适配器（5 V/ 3 A） • USB Type-C数据线	上述项目用于将Debian OS烧录到Micro-SD上。

注：
*: M：必须。O：可选

2.2. 准备内核镜像

按照以下步骤准备内核镜像：

1. 制作内核镜像 (第 8页)
2. 在昉·星光 2上更新内核镜像 (第 10页)

2.2.1. 制作内核镜像

请按照以下步骤制作内核镜像：

1. 拉取内核代码：

```
$ git clone https://github.com/starfive-tech/VisionFive2.git
```

提示：

- 更多操作信息请参见：https://github.com/starfive-tech/VisionFive2。
- 此步骤使用的Debian版本号为`starfive-jh7110-VF2_515_v2.5.0-69`

  链接： https://debian.starfivetech.com/

  对应的软件版本为`VisionFive2 Software v2.5.0`

  链接：https://github.com/starfive-tech/VisionFive2/releases/tag/VF2_v2.5.0

2. 编译代码：

   a. 进入到以下目录：

   `/VisionFive2/linux/arch/riscv/configs`

   b. 使用以下文件覆盖`starfive_visionfive2_defconfig`。

   docker_config_20230215

   注：
   下载此文件需要注册或登录RVspace。

   c. 在`/VisionFive2/work/linux`目录下运行脚本`./build_kernel_vf2.sh`

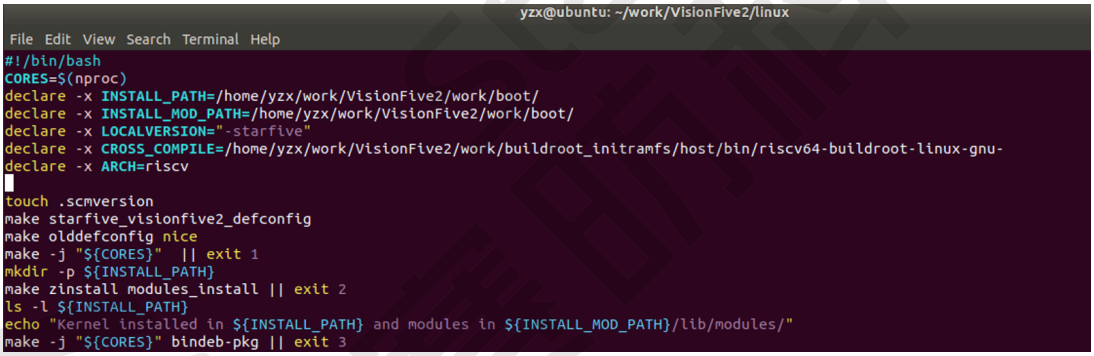
   注：

   - 脚本下载地址：build_kernel_vf2.sh

     注：
     下载此文件需要注册或登录RVspace。

   - 需要把脚本文件的目标路径改成自己的文件路径。

图 2-1 示例运行过程



3. 确认编译后生成的文件：

   - `./VisionFive2/work/boot`路径下：
     - `config-5.15.0-starfive-dirty`
     - `System.map-5.15.0-starfive-dirty`
     - `vmlinuz-5.15.0-starfive-dirty`

     注：
     为简化后续操作，重命名上述生成的文件为：
     - `config-5.15.0-starfive`
     - `System.map-5.15.0-starfive`
     - `vmlinuz-5.15.0-starfive`

   - `./VisionFive2/work`路径下：
     - `linux-headers-5.15.0-starfive_5.15.0-starfive-1_riscv64.deb`
     - `linux-image-5.15.0-starfive_5.15.0-starfive-1_riscv64.deb`
     - `linux-libc-dev_5.15.0-starfive-1_riscv64.deb`

2.2.2. 在昉·星光 2上更新内核镜像

请按照以下步骤在昉·星光 2上更新内核镜像。

1. 烧录Debian镜像至SD卡，需在昉·星光 2上正常启动一次(必要操作)。

   提示：

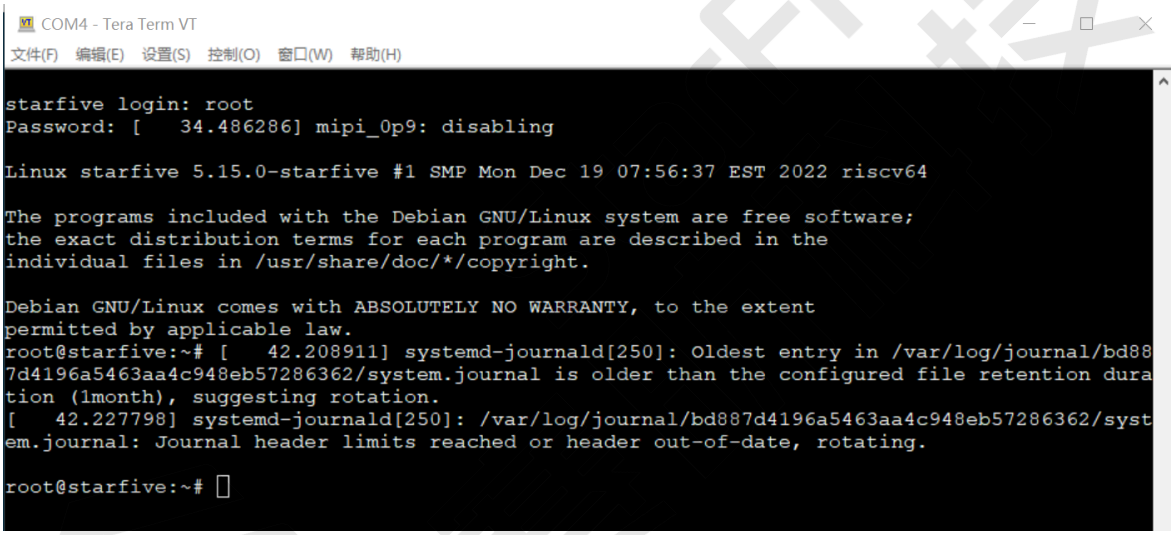
   - 具体烧录步骤参见《昉·星光 2 单板计算机快速参考手册》中的将*OS*烧录到*Micro-SD*卡上章节。
   - 此步骤使用的Debian版本号为`starfive-jh7110-VF2_515_v2.5.0-69`

     链接： https://debian.starfivetech.com/

     对应的软件版本为`VisionFive2 Software v2.5.0`

     链接：https://github.com/starfive-tech/VisionFive2/releases/tag/VF2_v2.5.0

   图 2-2 昉·星光 2启动输出示例

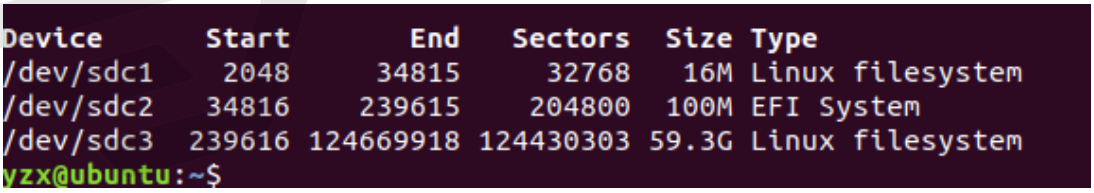
   

2. Ubuntu开发环境下插入SD卡并进行Mount分区操作:

   a. 列出设备:

   ```
   fdisk -l
   ```

   图 2-3 示例输出

   b. 执行以下命令进行Mount操作:

   ```
   sudo mount /dev/sdc2 /media/<Username>/root/boot
   ```

   示例命令：

   ```
   sudo mount /dev/sdc2 /media/yzx/root/boot
   ```

3. 执行以下命令，拷贝生成的Kernel文件至`/media/<Username>/root/boot/boot`:

   ```
   sudo cp boot/vmlinuz-5.15.0-starfive /media/<Username>/root/boot/boot && sync
   sudo cp boot/config-5.15.0-starfive /media/<Username>/root/boot/boot && sync
   ```

```
sudo cp boot/System.map-5.15.0-starfive /media/<Username>/root/boot/boot && sync
```

4. 执行以下命令，拷贝deb镜像包至 `/media/<Username>/root/usr`：

```
sudo cp linux-image-5.15.0-starfive_5.15.0-starfive-1_riscv64.deb /media/<<Username>/ root/usr/ && sync
sudo cp linux-libc-dev_5.15.0-starfive-1_riscv64.deb /media/<Username>/root/usr/ && sync
sudo cp linux-headers-5.15.0-starfive_5.15.0-starfive-1_riscv64.deb /media/<Username>/root/usr/ && sync
```

5. 下载Docker离线安装包，并拷贝至`/media/<username>/root/usr`。

   提示：
   安装包下载路径：https://github.com/carlosedp/riscv-bringup/releases/download/v1.0/docker-v20.10.2-dev_riscv64.deb

6. 将SD卡插入昉·星光 2并重新启动，会进入紧急模式，输入密码`starfive`即可登录。
7. 将SD卡先拔出再重新插入至昉·星光 2，登录成功后，进入`/usr`目录下安装deb包。

   注：
   需要按照命令顺序安装。

```
dpkg -i linux-headers-5.15.0-starfive_5.15.0-starfive-1_riscv64.deb
dpkg -i linux-libc-dev_5.15.0-starfive-1_riscv64.deb
dpkg -i linux-image-5.15.0-starfive_5.15.0-starfive-1_riscv64.deb
```

8. 重启系统，进入正常登录界面，输入帐号及密码。
9. 执行以下命令，安装Docker相关依赖包和Docker离线包：

```
apt install libip6tc2 libyajl2 conntrack ebtables ethtool iptables socat libyajl-dev
dpkg –i docker-v20.10.2-dev_riscv64.deb
```

3. 在昉·星光 2上启动Docker

请按照以下步骤在昉·星光 2上启动Docker:

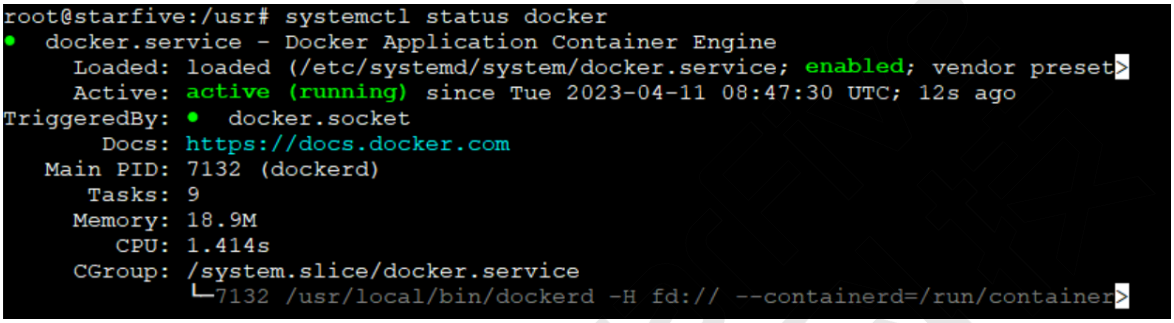
1. 按顺序执行以下命令:

```
dockerd
systemctl start docker
```

2. 执行以下命令:

```
systemctl status docker
```

图 3-1 示例输出

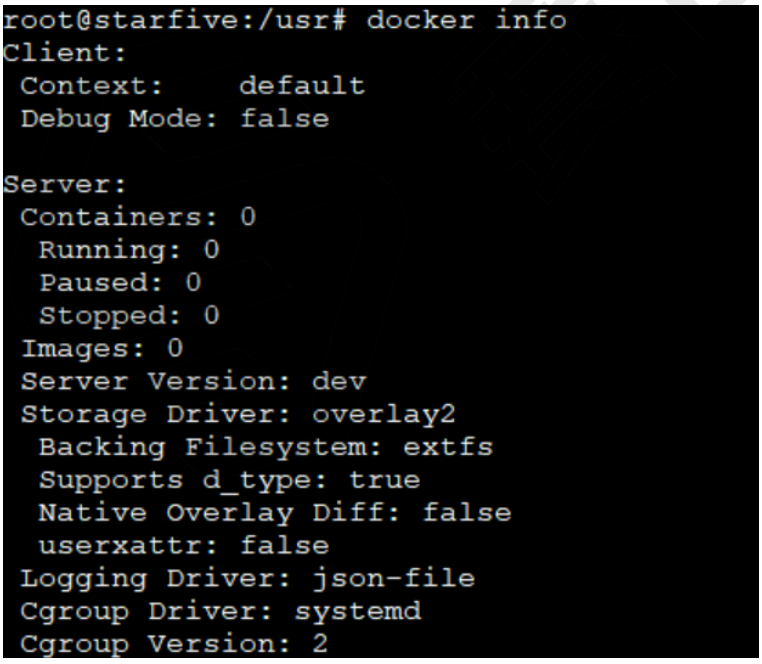


注:
若输出结果不是`enabled`、`Active: active (running)`，建议重启一次系统再执行此步骤。

3. 执行以下命令:

```
docker info
```

图 3-2 示例输出



4. 执行以下命令，运行`hello-world`:

```
docker run --rm hello-world
```

图 3-3 示例输出

```
root@starfive:/usr# docker run --rm hello-world
[ 1460.997698] docker0: port 1(veth69587cd) entered blocking state
[ 1461.003723] docker0: port 1(veth69587cd) entered disabled state
[ 1461.009999] device veth69587cd entered promiscuous mode
[ 1462.529926] eth0: renamed from veth8822f9d
[ 1462.619834] docker0: port 1(veth69587cd) entered blocking state
[ 1462.625792] docker0: port 1(veth69587cd) entered forwarding state

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (riscv64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

5. 下载RISC-V版本的Alpine Docker镜像并运行：

```
docker pull riscv64/alpine:edge
docker run -it alpine:edge
```

图 3-4 示例命令与输出

```
root@starfive:/usr# docker pull riscv64/alpine:edge
edge: Pulling from riscv64/alpine
1c9566da74e4: Pull complete
Digest: sha256:3894e4e3ea0345d0627776199362bf3e68a057a2786b253d8588cf68220f7de3
Status: Downloaded newer image for riscv64/alpine:edge
docker.io/riscv64/alpine:edge
root@starfive:/usr# docker run -it  alpine:edge
Unable to find image 'alpine:edge' locally
edge: Pulling from library/alpine
Digest: sha256:2d01a16bab53a8405876cec4c27235d47455a7b72b75334c614f2fb0968b3f90
Status: Downloaded newer image for alpine:edge
[ 1582.673732] docker0: port 1(veth6116cd8) entered blocking state
[ 1582.679750] docker0: port 1(veth6116cd8) entered disabled state
[ 1582.686311] device veth6116cd8 entered promiscuous mode
[ 1584.340633] eth0: renamed from veth31f1453
[ 1584.390567] docker0: port 1(veth6116cd8) entered blocking state
[ 1584.396526] docker0: port 1(veth6116cd8) entered forwarding state
/ # ls
bin    etc    lib    mnt    proc   run    srv    tmp    var
dev    home   media  opt    root   sbin   sys    usr
/ #
```

6. 在Docker镜像中执行命令来验证：

```
cat /etc/os-release
ifconfig
arp -a
ping <Website>
```

提示：
执行命令前需要将<*Website*>替换为任意网址。

图 3-5 示例命令与输出

```
/ # cat  /etc/os-release
NAME="Alpine Linux"
ID=alpine
VERSION_ID=3.18_alpha20230329
PRETTY_NAME="Alpine Linux edge"
HOME_URL="https://alpinelinux.org/"
BUG_REPORT_URL="https://gitlab.alpinelinux.org/alpine/aports/-/issues"
/ # ifconfig
eth0      Link encap:Ethernet  HWaddr 02:42:AC:11:00:02
          inet addr:172.17.0.2  Bcast:172.17.255.255  Mask:255.255.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

/ # arp -a
/ # ping www.baidu.com
PING www.baidu.com (180.101.50.242): 56 data bytes
64 bytes from 180.101.50.242: seq=0 ttl=51 time=7.881 ms
64 bytes from 180.101.50.242: seq=1 ttl=51 time=7.536 ms
```

如上输出显示，您已经成功在昉·星光 2上运行Docker了！