FreddieOliveira / **docker.md**

Last active yesterday • Report abuse

This tutorial shows how to run docker natively on Android, without VMs and chroot.

docker.md

Docker on Android 🐳📱

Edit 🎉

All packages, except for Tini have been added to termux-root. To install them, simply `pkg install root-repo && pkg install docker` . This will install the whole docker suite, left only Tini to be compiled manually.

Summary

1. Intro

This tutorial presents a step by step guide on how to run docker containers directly on Android. By directly I mean there's no VM involved nor chrooting inside a GNU/Linux rootfs. This is docker purely in Android. Yes, ***it is*** possible.

Bear in mind that you'll have to root your phone, mess with and compile your phone's kernel and docker suite. So, be prepared to get your hands dirty.

2. Building

2.1. Rooting

This step is pretty device specific, so there's no way to write a generic tutorial here. You'll need to google for instructions for your device and follow them.

Just be aware that you may lose your phone's warrant and all your data will be erased after unlocking the bootloader, so make a backup of your important stuff.

2.2. Kernel

2.2.1. General compiling instructions

Compiling the phone's kernel is also device specific, but some major tips may help you out.

First, google about instructions for your phone. Start by compiling the kernel without any modification. Flash it and hope for the best. If everything went well, then you can proceed to the modifications.

Note that flashing the kernel won't erase any data in your phone. The worst that can happen is you get stuck in a boot loop. In this case, you can flash a kernel that's known to be working or just flash a working ROM, since it contains a kernel with it. None of these operations erase any data in your phone.

2.2.2. Modifications

Now that you (hopefully) are able to compile the kernel, let's talk about what matters. Docker needs a lot of features that are disabled by default in Android's kernel.

To check the necessary features list, first install the Termux app in your phone. This is the terminal emulator that we're going to use throughout this guide. It has a package manager with many tools compiled for Android.

Next, open Termux and download a script to check your kernel:

```
$ pkg install wget
$ wget https://raw.githubusercontent.com/moby/moby/master/contrib
/check-config.sh
$ chmod +x check-config.sh
$ sed -i '1s_.*_#!/data/data/com.termux/files/usr/bin/bash_' check-
config.sh
$ sudo ./check-config.sh
```

Now, in your computer, open the kernel's configuration menu. This menu is a modified version of dialog, a ncurses window menu, in which you can enable and disable the kernel features. To look for some item in particular, you can press the `/` key and type the item name and hit Enter. This will show the description and location of the item.

For now, we want to enable the `Generally Necessary` items, the `Network Drivers` items and some `Optional Features`. For the `Storage Drivers` we'll be using the `overlay`.

2.2.3. Patching

Before compiling the kernel there are two files that need to be patched.

kernel/Makefile

The first one is the `kernel/Makefile`. Although not strictly necessary to modify this file, it will help by making it possible to check if your kernel has all the necessary features docker needs.

If you do not apply this patch, the output of the `check-config.sh` script used above won't be reliable after recompiling the kernel.

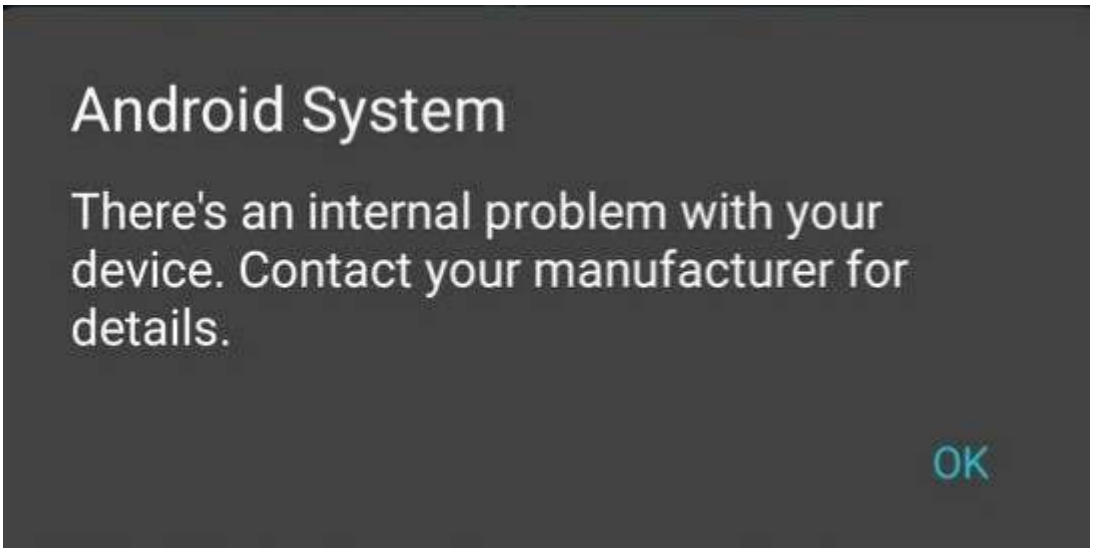
Check the patch at the attachments section and modify your Makefile accordingly.

net/netfilter/xt_qtaguid.c

This second file *needs to be patched* because of a bug introduced by Google. After you run any container, a seg fault will be generated due to a null pointer dereference and your phone will freeze and reboot. If you work at Google or know someone who does, warn him/her about it.

Check the patch at the attachments section and modify your xt_qtaguid.c accordingly.

Now that everything is setup, compile and flash the kernel. If you applied the Makefile patch, you'll see this warning everytime your phone boots:



Don't worry though, this is a harmless warning remembering you that you're using a modified kernel.

2.3. Docker

See Edit.

Once you have a supported kernel, it's time to compile the docker suite. It's a suite because it's not just one program, but rather a set of different programs that we'll need to compile separately. So hands on.

Firts, let's install the packages we're gonna use to build docker in Termux:

```
$ pkg install go make cmake ndk-multilib tsu
```

Now we're ready to start compiling things. Create a work directory where the packages will be downloaded and built:

```
$ mkdir $TMPDIR/docker-build
$ cd $TMPDIR/docker-build
```

Download all the patches files into there and let's begin. All commands for the differents packages that'll be compiled next is meant to be executed inside this folder.

2.3.1. dockercli

See Edit.

This is the docker client, which will talk to the docker daemon. This package will compile a binary named `docker` and all docker man pages. To build and install it:

```
$ cd $TMPDIR/docker-build
$ wget https://github.com/docker/cli/archive/v20.10.2.tar.gz -O
cli-20.10.2.tar.gz
$ tar xf cli-20.10.2.tar.gz
$ mkdir -p src/github.com/docker
$ mv cli-20.10.2 src/github.com/docker/cli
$ export GOPATH=$(pwd)
$ export VERSION=v20.10.2-ce
$ export DISABLE_WARN_OUTSIDE_CONTAINER=1
$ cd src/github.com/docker/cli
$ xargs sed -i 's_/var/\(run/docker\.sock\)_/data/docker/\1_g' <
<(grep -R /var/run/docker\.sock | cut -d':' -f1 | sort | uniq)
$ patch vendor/github.com/containerd/containerd/platforms/database.go
../../../../database.go.patch.txt
$ patch scripts/docs/generate-man.sh ../../../../generate-
man.sh.patch.txt
$ patch man/md2man-all.sh ../../../../md2man-all.sh.patch.txt
$ patch cli/config/config.go ../../../../config.go.patch.txt
$ make dynbinary
$ make manpages
$ install -Dm 0700 build/docker-android-* $PREFIX/bin/docker
```

```
$ install -Dm 600 -t $PREFIX/share/man/man1 man/man1/*
$ install -Dm 600 -t $PREFIX/share/man/man5 man/man5/*
$ install -Dm 600 -t $PREFIX/share/man/man8 man/man8/*
```

2.3.2. dockerd

See Edit.

The docker daemon is the most problematic binary that's gonna be compiled. It needs so many patches that's easier to modify the code in a batch with sed. Despite the need of modifying a lot of files, the modifications by themselfs are rather simple:

1. Substitute every occurrence of `runtime.GOOS` by the string `"linux"`;
2. Remove unneeded imports of the `runtime` lib.

By doing that, we are in essence spoofing our operating system as a Linux one: everytime the code would do the `runtime.GOOS == "linux"` comparison (which would become `"android" == "linux"`, and thus `false`) it will now do `"linux" == "linux"` and thus `true`.

To make the substitution across every file, we'll run a sed command. After that, some files will now give the extremely annoying unturnable-off go lang "feature" `imported and not used` error, because the only function these files were using from the `runtime` package was the `runtime.GOOS`. So, to fix it we'll use an horrible but simple solution: we'll keep trying to compile the code and at each failed attempt we'll fix the reported files till we get it to compile successfully.

```
$ cd $TMPDIR/docker-build
$ wget https://github.com/moby/moby/archive/v20.10.2.tar.gz -O
moby-20.10.2.tar.gz
$ tar xf moby-20.10.2.tar.gz
$ cd moby-20.10.2
$ export DOCKER_GITCOMMIT=8891c58a43
$ export DOCKER_BUILDTAGS='exclude_graphdriver_btrfs
exclude_graphdriver_devicemapper exclude_graphdriver_quota selinux
exclude_graphdriver_aufs'
$ patch cmd/dockerd/daemon.go ../daemon.go.patch
$ xargs sed -i "s_\(/etc/docker\)_$PREFIX\1_g" < <(grep -R
/etc/docker | cut -d':' -f1 | sort | uniq)
$ xargs sed -i 's_\(/run/docker/plugins\)_/data/docker\1_g' < <(grep
-R '/run/docker/plugins' | cut -d':' -f1 | sort | uniq)
$ xargs sed -i 's/[a-zA-Z0-9]*\.GOOS/"linux"/g' < <(grep -R '[a-zA-
Z0-9]*\.GOOS' | cut -d':' -f1 | sort | uniq)
$ (while ! IFS='' files=$(AUTO_GOPATH=1 PREFIX='' hack/make.sh
dynbinary 2>&1 1>/dev/null); do if ! xargs sed -i 's/\("runtime"\)/_
\1/' < <(echo $files | grep runtime | cut -d':' -f1 | cut -c38-);
then echo $files; exit 1; fi; done)
```

```
$ install -Dm 0700 bundles/dynbinary-daemon/dockerd $PREFIX/bin
/dockerd-dev
```

A binary called dockerd-dev was compiled and installed, but in order to it run correctly, the cgroups need to be mounted. Since Android mounts the cgroups in a non standard location we need to fix this. To do so, a script named dockerd will be created that will mount crgoups in the correct path if needed and call dockerd-dev next.

```
$ cat << "EOF" > $PREFIX/bin/dockerd
#!/data/data/com.termux/files/usr/bin/bash

export PATH="${PATH}:/system/xbin:/system/bin"
opts='rw,nosuid,nodev,noexec,relatime'
cgroups='blkio cpu cpuacct cpuset devices freezer memory pids
schedtune'

# try to mount cgroup root dir and exit in case of failure
if ! mountpoint -q /sys/fs/cgroup 2>/dev/null; then
  mkdir -p /sys/fs/cgroup
  mount -t tmpfs -o "${opts}" cgroup_root /sys/fs/cgroup || exit
fi

# try to mount cgroup2
if ! mountpoint -q /sys/fs/cgroup/cg2_bpf 2>/dev/null; then
  mkdir -p /sys/fs/cgroup/cg2_bpf
  mount -t cgroup2 -o "${opts}" cgroup2_root /sys/fs/cgroup/cg2_bpf
fi

# try to mount differents cgroups
for cg in ${cgroups}; do
  if ! mountpoint -q "/sys/fs/cgroup/${cg}" 2>/dev/null; then
    mkdir -p "/sys/fs/cgroup/${cg}"
    mount -t cgroup -o "${opts},${cg}" "${cg}" "/sys/fs/cgroup/${cg}"
\
    || rmdir "/sys/fs/cgroup/${cg}"
  fi
done

# start the docker daemon
$PREFIX/bin/dockerd-dev $@
EOF
```

Make the script executable:

```
$ chmod +x $PREFIX/bin/dockerd
```

And finally configure some dockerd options:

```
$ mkdir -p $PREFIX/etc/docker
$ cat << "EOF" > $PREFIX/etc/docker/daemon.json
{
    "data-root": "/data/docker/lib/docker",
    "exec-root": "/data/docker/run/docker",
    "pidfile": "/data/docker/run/docker.pid",
    "hosts": [
        "unix:///data/docker/run/docker.sock"
    ],
    "storage-driver": "overlay2"
}
EOF
```

> **Warning:** dockerd will store all its files, like containers, images, volumes, etc inside the `/data/docker` folder, which means you'll lose everything if you format the phone (flash a ROM). This folder was chosen instead of storing things inside Termux installation folder, because dockerd fails when setting up the overlay storage driver there. It seems Android mounts the `/data/data` folder with some options that prevent overlayfs to work, or the filesystem doesn't support it.

2.3.3. tini

tini is an optional dependency of dockerd in case you want the `init` process to be the first process of the container being ran (for this use the `--init` flag when creating a container). Having `init` as the parent of all other proccess ensures that a proper clean up inside the container is made regarding zombie processes. For a detailed explanation on its benefits and when to use it, check here: krallin/tini#8

```
$ cd $TMPDIR/docker-build
$ wget https://github.com/krallin/tini/archive/v0.19.0.tar.gz
$ tar xf v0.19.0.tar.gz
$ cd tini-0.19.0
$ mkdir build
$ cd build
$ cmake -DCMAKE_BUILD_TYPE=Release -DCMAKE_INSTALL_PREFIX=$PREFIX ..
$ make -j8
$ make install
$ ln -s $PREFIX/bin/tini-static $PREFIX/bin/docker-init
```

2.3.4. libnetwork

See Edit.

Another dockerd dependency needed when using the `-p` flag while creating a container:

```
$ cd $TMPDIR/docker-build
$ wget https://github.com/moby/libnetwork/archive
/448016ef11309bd67541dcf4d72f1f5b7de94862.tar.gz
$ tar xf 448016ef11309bd67541dcf4d72f1f5b7de94862.tar.gz
$ mkdir -p src/github.com/docker
$ mv libnetwork-448016ef11309bd67541dcf4d72f1f5b7de94862
src/github.com/docker/libnetwork
$ export GOPATH="$(pwd)"
$ cd src/github.com/docker/libnetwork
$ go build -o docker-proxy github.com/docker/libnetwork/cmd/proxy
$ strip docker-proxy
$ install -Dm 0700 docker-proxy $PREFIX/bin/docker-proxy
```

2.3.5. containerd

See Edit.

This is a dockerd dependency. Some patches are needed to fix path locations, build the manuals correctly and compile extra binaries used by dockerd that are not build by default by the Makefile:

```
$ cd $TMPDIR/docker-build
$ wget https://github.com/containerd/containerd/archive/v1.4.3.tar.gz
$ tar xf v1.4.3.tar.gz
$ mkdir -p src/github.com/containerd
$ mv containerd-1.4.3 src/github.com/containerd/containerd
$ export GOPATH=$(pwd)
$ cd src/github.com/containerd/containerd
$ xargs sed -i "s_\(/etc/containerd\)_$PREFIX\1_g" < <(grep -R
/etc/containerd | cut -d':' -f1 | sort | uniq)
$ patch runtime/v1/linux/bundle.go ../../../../bundle.go.patch.txt
$ patch runtime/v2/shim/util_unix.go ../../..
/../util_unix.go.patch.txt
$ patch Makefile ../../../../Makefile.patch
$ patch platforms/database.go ../../../../database.go.patch.txt
$ patch vendor/github.com/cpuguy83/go-md2man/v2/md2man.go ../../..
/../md2man.go.patch.txt
$ BUILDTAGS=no_btrfs make -j8
$ make -j8 man
$ DESTDIR=$PREFIX make install
$ DESTDIR=$PREFIX/share make install-man
```

Lastly, some configurations:

```
$ mkdir -p $PREFIX/etc/containerd
$ cat << "EOF" > $PREFIX/etc/containerd/config.toml
root = "/data/docker/var/lib/containerd"
state = "/data/docker/run/containerd"
imports = ["$PREFIX/etc/containerd/runtime_*.toml", "./debug.toml"]

[grpc]
  address = "/data/docker/run/containerd/containerd.sock"

[debug]
  address = "/data/docker/run/containerd/debug.sock"

[plugins]
  [plugins.opt]
    path = "/data/docker/opt"
  [plugins.cri.cni]
    bin_dir = "/data/docker/opt/cni/bin"
    conf_dir = "/data/docker/etc/cni/net.d"
EOF
```

> **Note:** unfortunately containerd files also can't be stored inside Termux installation folder, failing with an error when creating the socket it uses.

2.3.6. runc

See Edit.

runc is a dependency of containerd. Conveniently for us, it's already provided by Termux's repository. Install it by simply:

```
$ pkg install root-repo
$ pkg install runc
```

3. Running

Now comes the truth time. To run the containers, first we need to start the daemon manually. To do so, it's advisable to install a terminal multiplexer so you can run the daemon in one pane and the container in others panes:

```
$ pkg install tmux
```

In one pane start dockerd:

```
$ sudo dockerd --iptables=false
```

And in others panes you can run the containers:

```
$ sudo docker run hello-world
```

> **Note:** Teaching how to use tmux is out of the scope of this guide, you can find good tutorials on YouTube. If you don't wanna use a terminal multiplexer, you can run dockerd in the background instead, with `sudo dockerd &>/dev/null &`.

3.1. Caveats

3.1.1. Internet access

The two network drivers tested so far are `bridge` and `host`. Here's how to get each of them working.

bridge

This is the default netwok driver. If you don't specify a driver, this is the type of network you are creating. Bridge networks isolate the container network by editing the iptables rules and creating a network interface called `Docker0` that serves as a bridge. All containers created with the bridge driver will use this interface. This is analogous to creating a VLAN and running the containers inside it.

But, there's a catch in Android: iptables rules policy is different here than on a conventional GNU/Linux system (more info here). For the bridge driver to work, you'll have to manually edit the iptable by running;

```
$ sudo ip route add default via 192.168.1.1 dev wlan0
$ sudo ip rule add from all lookup main pref 30000
```

> **Note:** change 192.168.1.1 according to your gateway IP.

Unfortunately, this means that changing networks will require you to re-configure the rules again.

host

Using the host driver, means to remove network isolation between the container and the Docker host, and use the host's networking directly. This way, the container will use the same network interface as your device (e.g. wlan0) and thus will share the same IP address.

To use this driver give the `--net=host --dns=8.8.8.8` flags when running a container.

3.1.2. Shared volumes

An easy way to share folders and files between containers and the host is to use a shared volume. For example, using the `-v ~/Documents/docker-share:/root/docker-share` flag when running a container, will make the `~/Documents/docker-share` folder from the host to be accessible inside the container `/root/docker-share` folder.

But, when talking about Android, things seems to never be as easy and straightforward as expected. Due to Android file system encryption, if you `ls` the `/root/docker-share` folder inside the container you might see a bunch of random letters, numbers and symbols instead of the folders and files names:

```
# ls /root/docker-share
+2xKy7JIRrcGrCf+o6KSeB  T6BJkyIa5OedXNrSyRKLbB
cwoDh,Nzt1l,5BsKA4hH8D
2aHRCQEyK8yYiiK9PEI9SA  Ue39lJVm4kIxGrS1bV07zB
lEpWZhTY9dNqJxCu+GqBuA
5ZRDLfHMwyik6RMe,f0WPA  X+yGLxXSgwxbCsFGRXuczC
y4ZWVvVBBjcxSWlJ9conED
GljgSZK5gFr7D4Fk7BHNeB  X1ATNoqhp,,ZsKjFXqKFiA
I3N5j0R4zmaQPKCWwKBlxD  Yzi+KmovJmIYFOCHtDCXkB
```

and if you try to read or create a file inside the volume you might get the `Required key not available` error.

No definitive solution was discovered so far, but a workaround is to `cat` the files from within the host to give the container temporary access to them. You can cat an individual file by:

```
$ sudo cat ~/Documents/docker-share/file.pdf >/dev/null
```

or all of them by:

```
$ sudo find ~/Documents/docker-share -exec cat {} >/dev/null \;
```

3.2. GUI

Yes, it's possible to run GUI programs inside a container! There's basically two ways of accomplishing it in a simple manner:

3.2.1. X11 Forwarding

Description

This method has the advantage of not making necessary the installation nor configuration of any additional programs inside the container; all you'll have to do is to setup the X inside termux and share its sockets with the container.

This is advisable to be used when you intend to run various containers with GUI, since you'll only have to install and setup a VNC once in the host, instead of doing it for each container. This will save storage space and time.

Steps

The first step is to enable the X11 repository in termux, this will allow installation of graphical interface related programs, like the VNC server we'll be using.

```
$ pkg install x11-repo
```

Then install a VNC server in termux:

```
$ pkg install tigervnc
```

> **Note:** These installations steps need to be executed only once.

Now, just run it:

```
$ vncserver -noxstartup -localhost
```

> **Note:** It's advisable to pass the `-geometry HEIGHTxWEIGHT` flag substituting HEIGHT and WEIGHT by your phone's screen resolution or some multiple of it.

> **Note:** The very first time you run it, you'll be prompted to setup a password. Note that passwords are not visible when you are typing them and it's maximal length is 8 characters. If you don't wanna use a passwd, use the `-SecurityTypes none` flag.

If everything is okay, you will see this message:

```
New 'localhost:1 ()' desktop is localhost:1
```

It means that X (vnc) server is available on display 'localhost:1'. Finally, export the DISPLAY environment variable according to that value:

```
$ export DISPLAY=:1
```

Now that the VNC server is configured and running in the host, start the container sharing the X related files as volumes:

```
$ sudo docker run -ti \
    --net="host" \
    --dns="8.8.8.8" \
    -e DISPLAY=$DISPLAY \
    -v $TMPDIR/.X11-unix:/tmp/.X11-unix \
    -v $HOME/.Xauthority:/root/.Xauthority \
    ubuntu
```

> **Note:** If by any reason you forget to export the DISPLAY before starting the container, you can still export it from inside it.

You'll now be able to launch GUI programs from inside the container, e.g.:

```
# echo 'APT::Sandbox::User "root";' > /etc/apt/apt.conf
# apt update
# apt install x11-apps
# xeyes
```

To check the GUI, you'll need to install a VNC client app in your Android phone, like VNC Viewer (developed by RealVNC Limited). Unfortunately it's not open source, but it's a good and intuitive VNC client for Android.

> **Note:** There's also an open source alternative developed by @pelya called XServer XSDL, which will not be covered by this guide (for now).

After installing the VNC Viewer app, open it and setup a new connection using 127.0.0.1 (or localhost) as the IP, 5901 as the port (the port is calculated as 5900 + {display number}) and when/if prompted, type the password choosen when running vnctiger for the first time.

3.2.2. VNC server within the container

Description

This method is very similar to the previous, with the difference that the X server will be installed inside the container instead of in the termux host.

The advantages are:

1. you aren't changing your host system by installing softwares on it (like the VNC server);
2. security, since you won't be sharing your host's X (this is only relevant when you are not the one running the container).

The main disadvantage is that you'll need to install and config the VNC server for each container you'd run a GUI program, thus making these containers big and time consuming to setup.

Steps

First, start a container:

```
$ sudo docker run -ti \
    --net="host" \
    --dns="8.8.8.8" \
    ubuntu
```

Now, a VNC server needs to be installed and configured inside the container. You can choose between TigerVNC or x11vnc:

TigerVNC

The same VNC server used above in termux. To install it:

```
# echo 'APT::Sandbox::User "root";' > /etc/apt/apt.conf
# apt update
# apt install tigervnc-standalone-server
```

Next, start it with:

```
# vncserver -noxstartup -localhost -SecurityTypes none
```

Here we disabled password (`-SecurityTypes none`) because using it causes things to crash as described in this issue report TigerVNC/tigervnc#800

If everything is okay, you will see this message:

```
New 'localhost:1 (root)' desktop at :1 on machine localhost
```

Export the DISPLAY environment variable according to that value:

```
# export DISPLAY=:1
```

From now on, you can already run GUI programs and access them using the VNC Viewer client as already described in the end of X11 Forwarding steps.

x11vnc

Install the x11vnc and the virtual fake X (since x11vnc can't emulate a X11 by itself):

```
# echo 'APT::Sandbox::User "root";' > /etc/apt/apt.conf
# apt update
# apt install x11vnc xvfb
```

Now, start it:

```
# x11vnc -nopw -forever -noshm -create
```

If everything is okay, you will see this message:

```
The VNC desktop is:      localhost:0
PORT=5900
```

This will open a xterm terminal which can be acessed by the VNC Viewer client as already described in the end of X11 Forwarding steps. From that terminal you can open the desired GUI program.

3.3. Steam (work in progress)

I'm not talking about running the useless steam app for Android, but about running the Desktop version and play the games inside a docker container. Yes, you read it right, it's possible to play your Steam games on Android!

(ACTUALLY NOT YET, BECAUSE I DIDN'T MANAGE TO GET OPENGL TO WORK, THAT'S WHY THIS IS A WORK IN PROGRESS. TO CONTRIBUTE OR STAY UP TO DATE ABOUT THE PROGRESS CHECK ptitSeb/box86#249)

To do so, we'll use an awesome x86 emulator for ARM developed by @ptitSeb called box86.

But first, you need to enable `System V IPC` under `General Setup` in the kernel config and recompile it again. That's because the steam binary needs some semaphore functions and will crash in case it can't use them.

Next, we hit a problem: box86 can only be compiled by a 32 bit toolchain. But, in fact, this can be easily circumvented by using a 32 bit container:

```
$ sudo docker run -ti \
    --net="host" \
    --dns="8.8.8.8" \
    -e DISPLAY=$DISPLAY \
    -w /root \
    -v $TMPDIR/.X11-unix:/tmp/.X11-unix \
    -v $HOME/.Xauthority:/root/.Xauthority \
    --platform=linux/arm \
    arm32v7/ubuntu
```

> **Note:** if your system is 32 bit already (run `uname -m` to check), you don't need to specify the `--platform=linux/arm` flag and can simply use `ubuntu` instead of `arm32v7/ubuntu` .

Now that we are inside the container, let's install the tools we're gonna use, as well as the steam .deb installer:

```
# echo 'APT::Sandbox::User root;' >> /etc/apt/apt.conf
# apt update
# apt install wget binutils xterm libvdpau1 libappindicator1 libnm0
libdbusmenu-gtk4
```

Install steam:

```
# wget https://steamcdn-a.akamaihd.net/client/installer/steam.deb
# ar x steam.deb
# mkdir steam
# tar xf data.tar.xz -C steam
# find steam -type d -exec sh -c 'mkdir -p /${0#*/}' {} \;
# find steam \! -type d -exec sh -c 'mv $0 /${0#*/}' {} \;
# patch /usr/lib/steam/bin_steam.sh bin_steam.sh.patch
# rm -rf steam* *.tar* bin_steam.sh.patch
# steam
```

Steam will fail with a bunch of errors, but that's expected. The important thing is that it installed the necessary files under `~/.local/share/Steam` , one of them being the steam binary. Finish the installation by adding it to the path:

```
# ln -sf /root/.local/share/Steam/ubuntu12_32/steam /usr/bin/steam
```

Now, we need to install the i386 version of some libs required by steam. For this, we're going to download them directly from Ubuntu packages. That's because if we instead simply apt install them we would be getting the arm32 version.

4. Attachments

4.1. kernel patches

- kernel/Makefile

```
diff --git a/kernel/Makefile b/kernel/Makefile
index d5c1115..2dea801 100644
--- a/kernel/Makefile
+++ b/kernel/Makefile
@@ -121,7 +121,7 @@ $(obj)/configs.o: $(obj)/config_data.h
# config_data.h contains the same information as ikconfig.h but gzipped.
# Info from config_data can be extracted from /proc/config*
targets += config_data.gz
-$(obj)/config_data.gz: arch/arm64/configs/lavender_stock-defconfig FORC
+$(obj)/config_data.gz: $(KCONFIG_CONFIG) FORCE
    $(call if_changed,gzip)

    filechk_ikconfiggz = (echo "static const char kernel_config_data[] _
```

- net/netfilter/xt_qtaguid.c

```
--- orig/net/netfilter/xt_qtaguid.c     2020-05-12 12:13:14.000000000 +0
+++ my/net/netfilter/xt_qtaguid.c       2019-09-15 23:56:45.000000000 +0
@@ -737,7 +737,7 @@
{
        struct proc_iface_stat_fmt_info *p = m->private;
        struct iface_stat *iface_entry;
-       struct rtnl_link_stats64 dev_stats, *stats;
+       struct rtnl_link_stats64 *stats;
        struct rtnl_link_stats64 no_dev_stats = {0};
@@ -745,13 +745,8 @@
        current->pid, current->tgid, from_kuid(&init_user_ns, current_fs
        iface_entry = list_entry(v, struct iface_stat, list);
```

```
+       stats = &no_dev_stats;
-       if (iface_entry->active) {
-               stats = dev_get_stats(iface_entry->net_dev,
-                                     &dev_stats);
-       } else {
-               stats = &no_dev_stats;
-       }
        /*
         * If the meaning of the data changes, then update the fmtX
         * string.
```

4.2. docker-cli patches

- vendor/github.com/containerd/containerd/platforms/database.go
- scripts/docs/generate-man.sh
- man/md2man-all.sh
- cli/config/config.go

4.3. dockerd patches

- cmd/dockerd/daemon.go

4.4. containerd patches

- runtime/v1/linux/bundle.go
- runtime/v2/shim/util_unix.go
- Makefile
- platforms/database.go
- vendor/github.com/cpuguy83/go-md2man/v2/md2man.go

5. Aknowledgements

I'd like to thank the Termux Dev team for this wonderful app and @xeffyr for discovering about the bug in `net/netfilter/xt_qtaguid.c` and sharing the patch, as well as all the conversation we had here that led to docker finally working.

Also @yjwong, for figuring out how to use the bridge network driver.

6. Final notes

If you are a docker developer reading this, please consider adding an official support for Android. Look above the possibilities it opens for a smartphone. If you are not a docker developer, consider supporting this by showing interest here. If we annoy the devs enough, this may become official (of they may simply unsubscribe from the thread and let it rot in the Issues section ¯_(ツ)_/¯).

Load earlier comments...

zotona commented on Jun 20, 2022 • edited

@**FreddieOliveira** Is docker could run correctly without Tini?
I install udocker , because i dont have a root. images loads correctly, but when i run image I had an error:
Error: while extracting image layer

Colo-Thor commented on Jun 25, 2022

@**Morakhiyasaiyam** How to create /var folder，docker-compose use /var/run folder, i try in kernel init.rc

```
on init
          mkdir /var
```

and in magisk /data/adb/post-fs-data.d folder add script are don't work

Morakhiyasaiyam commented on Jun 25, 2022

> @**Morakhiyasaiyam** How to create /var folder，docker-compose use /var/run folder, i try in kernel init.rc `on init  mkdir /var`
>
> and in magisk /data/adb/post-fs-data.d folder add script are don't work

sudo mount -o remount,rw /
sudo mkdir -p /var/run/

I will post easy and improved tutorial with my fixes and some easy pissy tricks to working with docker on Android then this post tomorrow

Colo-Thor commented on Jun 25, 2022

> @**Morakhiyasaiyam** How to create /var folder，docker-compose use /var/run folder, i try in kernel init.rc `on init  mkdir /var`
> and in magisk /data/adb/post-fs-data.d folder add script are don't work

> sudo mount -o remount,rw / sudo mkdir -p /var/run/
>
> I will post easy and improved tutorial with my fixes and some easy pissy tricks to working with docker on Android then this post tomorrow

Thanks, i look forward to your articles

zotona commented on Jun 28, 2022

> **@Morakhiyasaiyam** How to create /var folder，docker-compose use /var/run folder, i try in kernel init.rc `on init  mkdir /var`
> and in magisk /data/adb/post-fs-data.d folder add script are don't work
>
> sudo mount -o remount,rw / sudo mkdir -p /var/run/
>
> I will post easy and improved tutorial with my fixes and some easy pissy tricks to working with docker on Android then this post tomorrow

we are waiting!))

Morakhiyasaiyam commented on Jun 29, 2022

> > **@Morakhiyasaiyam** How to create /var folder，docker-compose use /var/run folder, i try in kernel init.rc `on init  mkdir /var`
> > and in magisk /data/adb/post-fs-data.d folder add script are don't work
>
> sudo mount -o remount,rw / sudo mkdir -p /var/run/
> I will post easy and improved tutorial with my fixes and some easy pissy tricks to working with docker on Android then this post tomorrow
>
> we are waiting!))

Yes i was little busy i will post today noon

Morakhiyasaiyam commented on Jun 29, 2022

@Colo-Thor @zotona
you can now check this repositery on github with detailed and updated guide and also refer this existing guide also
https://github.com/Morakhiyasaiyam/Docker-native-on-Termux-on-Android
thanks

Kris013f commented on Jul 30, 2022 • edited

My kernel does not include the CONFIG_AUFS_FS option at all.
The kernel stops building after enabling CONFIG_SECURITY_APPARMOR.
Unfortunately, I don't have the net/netfilter/xt_qtaguid.c file: Kernel , Sony Xperia XA2 Pioneer
$: sudo docker run hello-world # reboots my phone on termux

I found information that from Android 9 and Kernel 4.9 xt_qtaguid is deprecated and replaced with xt_bpf

After compiling the kernel, I have:

```
# I'm building kernel. On Android via Magisk, I patching boot.img
# $ adb reboot fastboot && fastboot boot ~/kernel/06_magisk_patched-25100_WJlqV.img
# $ adb root; adb shell /data/data/com.termux/files/home/dockerKernel/check-
config.sh

Generally Necessary:
- cgroup hierarchy: cgroupv2
  Controllers:
  - cpu: missing
  - cpuset: missing
  - io: missing
  - memory: missing
  - pids: available
  ...
- CONFIG_SECURITY_APPARMOR: missing
...
- Storage Drivers:
  - "aufs":
    - CONFIG_AUFS_FS: missing
...
  - "zfs":
    - /dev/zfs: missing
    - zfs command: missing
    - zpool command: missing
# others options is enabled
```

On two terminals in Termux via ssh -p 8022 ...:

```
sudo dockerd                  # or ns && ds aliases run ok in one terminal
sudo docker pull hello-world  # is ok to on the second terminal
sudo docker run hello-world   # this restarts my phone and the logs below
```

Log docker:

```
~ $ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
~ $ docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
Digest: sha256:53f1bbee2f52c39e41682ee1d388285290c5c8a76cc92b42687eecf38e0af3f0
Status: Image is up to date for hello-world:latest
docker.io/library/hello-world:latest
~ $ docker ps -a
```

```
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
~ $ docker run hello-world
docker: Error response from daemon: OCI runtime create failed:
container_linux.go:370: starting container process caused: process_linux.go:459:
container init caused: rootfs_linux.go:59: mounting "mqueue" to rootfs at
"/dev/mqueue" caused: device or resource busy: unknown.
ERRO[0000] error waiting for container: context canceled
```

Log dockerd:

```
...
INFO[2022-07-30T19:21:54.727900612Z] Docker daemon
commit=aa7e414 graphdriver(s)=overlay2 version=dev
INFO[2022-07-30T19:21:54.731618164Z] Daemon has completed initialization
INFO[2022-07-30T19:21:54.826863373Z] API listen on /data/docker/run/docker.sock
INFO[2022-07-30T19:22:27.517670287Z] /etc/resolv.conf does not exist
INFO[2022-07-30T19:22:27.517791277Z] No non-localhost DNS nameservers are left in
resolv.conf. Using default external servers: [nameserver 8.8.8.8 nameserver 8.8.4.4]
INFO[2022-07-30T19:22:27.517823881Z] IPv6 enabled; Adding default IPv6 external
servers: [nameserver 2001:4860:4860::8888 nameserver 2001:4860:4860::8844]
time="2022-07-30T19:22:27.619591225Z" level=info msg="starting signal loop"
namespace=moby path=/data/docker/run/docker/containerd/daemon
/io.containerd.runtime.v2.task
/moby/4351e7ab82dee1f247e8828c0b389cfef5e98ff8a846915edbd334ed05983d21 pid=14366
```

Any advice?

Thanks for a good job

Morakhiyasaiyam commented on Aug 1, 2022 • edited

> My kernel does not include the CONFIG_AUFS_FS option at all. The kernel stops building after enabling CONFIG_SECURITY_APPARMOR. Unfortunately, I don't have the net/netfilter /xt_qtaguid.c file: Kernel , Sony Xperia XA2 Pioneer $: sudo docker run hello-world # reboots my phone on termux
>
> I found information that from Android 9 and Kernel 4.9 xt_qtaguid is deprecated and replaced with xt_bpf
>
> After compiling the kernel, I have:
>
> ```
> # I'm building kernel. On Android via Magisk, I patching boot.img
> # $ adb reboot fastboot && fastboot boot ~/kernel/06_magisk_patched-
> 25100_WJlqV.img
> # $ adb root; adb shell /data/data/com.termux/files/home/dockerKernel/check-
> config.sh
>
> Generally Necessary:
> - cgroup hierarchy: cgroupv2
>   Controllers:
>   - cpu: missing
> ```

```
    - cpuset: missing
    - io: missing
    - memory: missing
    - pids: available
    ...
  - CONFIG_SECURITY_APPARMOR: missing
  ...
  - Storage Drivers:
    - "aufs":
      - CONFIG_AUFS_FS: missing
  ...
    - "zfs":
      - /dev/zfs: missing
      - zfs command: missing
      - zpool command: missing
  # others options is enabled
```

On two terminals in Termux via ssh -p 8022 ...:

```
sudo dockerd                  # or ns && ds aliases run ok in one terminal
sudo docker pull hello-world  # is ok to on the second terminal
sudo docker run hello-world   # this restarts my phone and the logs below
```

Log docker:

```
~ $ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
~ $ docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
Digest: sha256:53f1bbee2f52c39e41682ee1d388285290c5c8a76cc92b42687eecf38e0af3f0
Status: Image is up to date for hello-world:latest
docker.io/library/hello-world:latest
~ $ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
~ $ docker run hello-world
docker: Error response from daemon: OCI runtime create failed:
container_linux.go:370: starting container process caused:
process_linux.go:459: container init caused: rootfs_linux.go:59: mounting
"mqueue" to rootfs at "/dev/mqueue" caused: device or resource busy: unknown.
ERRO[0000] error waiting for container: context canceled
```

Log dockerd:

```
...
INFO[2022-07-30T19:21:54.727900612Z] Docker daemon
commit=aa7e414 graphdriver(s)=overlay2 version=dev
INFO[2022-07-30T19:21:54.731618164Z] Daemon has completed initialization
INFO[2022-07-30T19:21:54.826863373Z] API listen on /data/docker/run/docker.sock
INFO[2022-07-30T19:22:27.517670287Z] /etc/resolv.conf does not exist
INFO[2022-07-30T19:22:27.517791277Z] No non-localhost DNS nameservers are left
in resolv.conf. Using default external servers: [nameserver 8.8.8.8 nameserver
8.8.4.4]
```

> ```
> INFO[2022-07-30T19:22:27.517823881Z] IPv6 enabled; Adding default IPv6 external
> servers: [nameserver 2001:4860:4860::8888 nameserver 2001:4860:4860::8844]
> time="2022-07-30T19:22:27.619591225Z" level=info msg="starting signal loop"
> namespace=moby path=/data/docker/run/docker/containerd/daemon
> /io.containerd.runtime.v2.task
> /moby/4351e7ab82dee1f247e8828c0b389cfef5e98ff8a846915edbd334ed05983d21
> pid=14366
> ```
>
> Any advice?
>
> Thanks for a good job

aufs and apparmor are not necessarily required,you can use docker without those options perfectly, So compile kernel without those options enabled and try again

Kris013f commented on Aug 8, 2022

I did just that. The above logs (Log docker, Log dockerd) are without the CONFIG_SECURITY_APPARMOR, CONFIG_AUFS_FS options set.
In Termux Android, the command "sudo docker run hello-world" restarts the phone. I found information that from Android 9 and Kernel 4.9 xt_qtaguid is deprecated and replaced with xt_bpf

Cyberavater commented on Aug 17, 2022

@**FreddieOliveira** Can you make a Magisk module to patch the kernel?

Kris013f commented on Aug 21, 2022

Yes, my kernel is patched by Magisk to have root access. It seems to me that the problem is the net/netfilter/xt_qtaguid.c file, which is not in the new kernels.

STUkh commented on Aug 26, 2022

Guide doesn't work for me even if kernel is patched. LG v60. On container run getting kernel panic and reboot. If anyone have ideas how to debug/trace issue or other patches - would be great

Dricecrushme commented on Oct 4, 2022

@github_doc_commu_copyright
Apk built in provides.

Dricecrushme commented on Oct 4, 2022

> @github_doc_commu_copyright Apk built in provides.

@Dricecrushme

FlotingDream commented on Oct 6, 2022 • edited

Work fine! thx for the amazing tutorial!
This is my patched Kernel -- Full natively support the Docker in Raspberry Pi 4 LOS 19.1 Android.
In https://github.com/FlotingDream/Raspberry-Pi-4-LOS-19.1-Docker
:)

Dricecrushme commented on Oct 9, 2022

```
Go1
```

unsureman commented on Nov 7, 2022

Someone managed to solve the

`Required key not available` Problem?

Currently I‘m running a container with a docker-compose.yml and create a volume with:

```
volume:
- „./hostfolder:/container/folder:z“
```

Everything works fine besides creating folders in some directories, or mv/cp files. I receive
`Required key not available`

Example

```
$ sudo docker exec -it mycontainer bash
# mkdir -p /container/folder/new
  –> working as intended
# mkdir -p /container/folder/new/new1
  –> Required key not available
```

lawal1232 commented on Nov 14, 2022

Can someone please help me to modify the kernel?
This documentation is very good but the part how to modify the kernel is missing.
I am using this kernel for my samsung s7:
https://github.com/TheGalaxyProject/tgpkernel-s7-o

After runing `sudo ./check-config.sh` I get alot of missing Configs.

MRColorR commented on Nov 16, 2022 • edited

@**FreddieOliveira** maybe i'm missing something, but why not add all the needed kernel config and patching to a repo with a manifest? This could then be added to the local_manifest.xml when building a Kernel or ROM repo syncing all the necessary files. This way adding docker support to any custom build ROM will be really easy. this could be similar to what wireguard done to introduce wireguard support in android kernel https://git.zx2c4.com/android_kernel_wireguard/about/ .

daoudeddy commented on Dec 12, 2022

GUIDE

Running `dockerd` as a Termux service:

Install termux-services:

```
pkg install termux-services
```

and then restart termux so that the service-daemon is started.

Create dockerd service:

```
mkdir -p $PREFIX/var/service/dockerd/log

ln -sf $PREFIX/share/termux-services/svlogger $PREFIX/var/service/dockerd/log/run

echo -e '#!/data/data/com.termux/files/usr/bin/sh\nexec sudo dockerd 2>&1' >
$PREFIX/var/service/dockerd/run
```

Run dockerd service

```
sv up dockerd
```

Stop dockerd service

```
sv down dockerd
```

Check the logs:

```
cat $PREFIX/var/log/sv/dockerd/current
```

Reference:
Termux-services Wiki

romanovj commented on Dec 17, 2022

I can't understand this at all

patch vendor/github.com/containerd/containerd/platforms/database.go ../../../../database.go.patch.txt
patch scripts/docs/generate-man.sh ../../../../generate-man.sh.patch.txt
patch man/md2man-all.sh ../../../../md2man-all.sh.patch.txt
patch cli/config/config.go ../../../../config.go.patch.txt

daoudeddy commented on Dec 18, 2022

> I can't understand this at all
>
> patch vendor/github.com/containerd/containerd/platforms/database.go ../../..
> /../database.go.patch.txt patch scripts/docs/generate-man.sh ../../../../generate-man.sh.patch.txt
> patch man/md2man-all.sh ../../../../md2man-all.sh.patch.txt patch cli/config/config.go ../../..
> /../config.go.patch.txt

You dont jave to do this anymore, docker is already in the root repo of termux, just run

```
pkg in root-repo
pkg in docker
```

neoterm-extra commented on Dec 19, 2022 • edited

我解决了docker是armv8l的问题
问题出在magisk
我重写了tsu和sudo

tsu

SU_LOCATION="/sbin/su"
SU_BASH="$TMPDIR/su.bash"

function main() {
local cmd

```
  while [[ "$#" > 0 ]]; do
      local arg="$1"; shift
      case "$arg" in
          "-h" | "--help" ) echo "Usage: su [-c commands]"; exit 0 ;;
          "-c" | "--command" ) cmd="$@"; break ;;
      esac
  done

  local su_info=$($SU_LOCATION --help 2>&1)
  local supportCmd=$(echo $su_info | grep "\-c\\b")
```

```
  local CURRENT_PWD="$(pwd)";
  local wd=${CURRENT_PWD//\"/\\\"}
  echo "# su.bash" >$SU_BASH
  echo "export HOME=$HOME/.suroot" >>$SU_BASH
  echo "export TERM=$TERM" >>$SU_BASH
  echo "export PREFIX=$PREFIX" >>$SU_BASH
  echo "export TMPDIR=$TMPDIR" >>$SU_BASH
  echo "export SHELL=$SHELL" >>$SU_BASH
  echo "export PATH=$PATH" >>$SU_BASH
  echo "export LD_LIBRARY_PATH=$LD_LIBRARY_PATH" >>$SU_BASH
  echo "cd \"$wd\"" >>$SU_BASH
  echo "export PWD=\"$wd\"" >>$SU_BASH
  if [[ $cmd == "" ]]; then
      echo "exec $SHELL" >>$SU_BASH
  else
      echo "exec $SHELL -c '$cmd'" >>$SU_BASH
  fi
  chmod 700 $SU_BASH

  if [[ $supportCmd ]]; then
      cmd="-c $SU_BASH"
  else
      cmd="0 $SU_BASH"
  fi

  $SU_LOCATION $cmd


}

main "$@"
```

sudo

#!/data/data/com.termux/files/usr/bin/bash

/data/data/com.termux/files/usr/bin/tsu -c "$@"

neoterm-extra commented on Dec 19, 2022

> ## 我解决了docker是armv8l的问题
>
> 问题出在magisk
> 我重写了tsu和sudo
>
> ## tsu

SU_LOCATION="/sbin/su" SU_BASH="$TMPDIR/su.bash"

function main() { local cmd

```
  while [[ "$#" > 0 ]]; do
      local arg="$1"; shift
      case "$arg" in
          "-h" | "--help" ) echo "Usage: su [-c commands]"; exit 0 ;;
          "-c" | "--command" ) cmd="$@"; break ;;
      esac
  done

  local su_info=$($SU_LOCATION --help 2>&1)
  local supportCmd=$(echo $su_info | grep "\-c\\b")

  local CURRENT_PWD="$(pwd)";
  local wd=${CURRENT_PWD//\"/\\\"}
  echo "# su.bash" >$SU_BASH
  echo "export HOME=$HOME/.suroot" >>$SU_BASH
  echo "export TERM=$TERM" >>$SU_BASH
  echo "export PREFIX=$PREFIX" >>$SU_BASH
  echo "export TMPDIR=$TMPDIR" >>$SU_BASH
  echo "export SHELL=$SHELL" >>$SU_BASH
  echo "export PATH=$PATH" >>$SU_BASH
  echo "export LD_LIBRARY_PATH=$LD_LIBRARY_PATH" >>$SU_BASH
  echo "cd \"$wd\"" >>$SU_BASH
  echo "export PWD=\"$wd\"" >>$SU_BASH
  if [[ $cmd == "" ]]; then
      echo "exec $SHELL" >>$SU_BASH
  else
      echo "exec $SHELL -c '$cmd'" >>$SU_BASH
  fi
  chmod 700 $SU_BASH

  if [[ $supportCmd ]]; then
      cmd="-c $SU_BASH"
  else
      cmd="0 $SU_BASH"
  fi

  $SU_LOCATION $cmd


}
```

main "$@"

sudo

#!/data/data/com.termux/files/usr/bin/bash /data/data/com.termux/files/usr/bin/tsu -c "$@"

如果你觉得这对你来说有帮助请来star我的github
(/doge)

owen31302 commented last week

I have passed the Generally Necessary section but I still running into issue when I start the docker daemon.
Anyone encounter the same issue?

```
~ $ sudo ./check-config.sh
info: reading kernel config from /proc/config.gz ...

Generally Necessary:
- cgroup hierarchy: properly mounted [/dev]
- CONFIG_NAMESPACES: enabled
- CONFIG_NET_NS: enabled
- CONFIG_PID_NS: enabled
- CONFIG_IPC_NS: enabled
- CONFIG_UTS_NS: enabled
- CONFIG_CGROUPS: enabled
- CONFIG_CGROUP_CPUACCT: enabled
- CONFIG_CGROUP_DEVICE: enabled
- CONFIG_CGROUP_FREEZER: enabled
- CONFIG_CGROUP_SCHED: enabled
- CONFIG_CPUSETS: enabled
- CONFIG_MEMCG: enabled
- CONFIG_KEYS: enabled
- CONFIG_VETH: enabled
- CONFIG_BRIDGE: enabled
- CONFIG_BRIDGE_NETFILTER: enabled (as module)
- CONFIG_IP_NF_FILTER: enabled
- CONFIG_IP_NF_TARGET_MASQUERADE: enabled
- CONFIG_NETFILTER_XT_MATCH_ADDRTYPE: enabled
- CONFIG_NETFILTER_XT_MATCH_CONNTRACK: enabled
- CONFIG_NETFILTER_XT_MATCH_IPVS: enabled
- CONFIG_NETFILTER_XT_MARK: enabled
- CONFIG_IP_NF_NAT: enabled
- CONFIG_NF_NAT: enabled
- CONFIG_POSIX_MQUEUE: enabled
- CONFIG_DEVPTS_MULTIPLE_INSTANCES: enabled
- CONFIG_NF_NAT_IPV4: enabled
- CONFIG_NF_NAT_NEEDED: enabled

Optional Features:
- CONFIG_USER_NS: enabled
- CONFIG_SECCOMP: enabled
- CONFIG_SECCOMP_FILTER: enabled
- CONFIG_CGROUP_PIDS: missing
- CONFIG_MEMCG_SWAP: missing
- CONFIG_MEMCG_SWAP_ENABLED: missing
- CONFIG_MEMCG_KMEM: missing
- CONFIG_RESOURCE_COUNTERS: enabled
- CONFIG_IOSCHED_CFQ: enabled
- CONFIG_CFQ_GROUP_IOSCHED: missing
- CONFIG_BLK_CGROUP: missing
- CONFIG_BLK_DEV_THROTTLING: missing
- CONFIG_CGROUP_PERF: missing
- CONFIG_CGROUP_HUGETLB: missing
- CONFIG_NET_CLS_CGROUP: missing
```

```
- CONFIG_CGROUP_NET_PRIO: missing
- CONFIG_CFS_BANDWIDTH: missing
- CONFIG_FAIR_GROUP_SCHED: enabled
- CONFIG_RT_GROUP_SCHED: enabled
- CONFIG_IP_NF_TARGET_REDIRECT: enabled
- CONFIG_IP_VS: enabled
- CONFIG_IP_VS_NFCT: missing
- CONFIG_IP_VS_PROTO_TCP: missing
- CONFIG_IP_VS_PROTO_UDP: missing
- CONFIG_IP_VS_RR: missing
- CONFIG_SECURITY_SELINUX: enabled
- CONFIG_SECURITY_APPARMOR: missing
- CONFIG_EXT3_FS: enabled
- CONFIG_EXT3_FS_XATTR: enabled
- CONFIG_EXT3_FS_POSIX_ACL: missing
- CONFIG_EXT3_FS_SECURITY: missing
    (enable these ext3 configs if you are using ext3 as backing filesystem)
- CONFIG_EXT4_FS: enabled
- CONFIG_EXT4_FS_POSIX_ACL: missing
- CONFIG_EXT4_FS_SECURITY: enabled
    enable these ext4 configs if you are using ext4 as backing filesystem
- Network Drivers:
  - "overlay":
    - CONFIG_VXLAN: missing
    - CONFIG_BRIDGE_VLAN_FILTERING: missing
      Optional (for encrypted networks):
      - CONFIG_CRYPTO: enabled
      - CONFIG_CRYPTO_AEAD: enabled
      - CONFIG_CRYPTO_GCM: missing
      - CONFIG_CRYPTO_SEQIV: enabled
      - CONFIG_CRYPTO_GHASH: missing
      - CONFIG_XFRM: enabled
      - CONFIG_XFRM_USER: enabled
      - CONFIG_XFRM_ALGO: enabled
      - CONFIG_INET_ESP: enabled
      - CONFIG_INET_XFRM_MODE_TRANSPORT: enabled
  - "ipvlan":
    - CONFIG_IPVLAN: missing
  - "macvlan":
    - CONFIG_MACVLAN: missing
    - CONFIG_DUMMY: enabled
  - "ftp,tftp client in container":
    - CONFIG_NF_NAT_FTP: enabled
    - CONFIG_NF_CONNTRACK_FTP: enabled
    - CONFIG_NF_NAT_TFTP: enabled
    - CONFIG_NF_CONNTRACK_TFTP: enabled
- Storage Drivers:
  - "aufs":
    - CONFIG_AUFS_FS: missing
  - "btrfs":
    - CONFIG_BTRFS_FS: missing
    - CONFIG_BTRFS_FS_POSIX_ACL: missing
  - "devicemapper":
    - CONFIG_BLK_DEV_DM: enabled
    - CONFIG_DM_THIN_PROVISIONING: missing
  - "overlay":
    - CONFIG_OVERLAY_FS: missing
  - "zfs":
    - /dev/zfs: missing
```

```
      - zfs command: missing
      - zpool command: missing

  Limits:
  - /proc/sys/kernel/keys/root_maxkeys: 1000000
```

Here is the error when I run the command:

```
~ $ sudo dockerd --iptables=false
mount: 'cgroup2_root'->'/sys/fs/cgroup/cg2_bpf': No such device
mount: 'blkio'->'/sys/fs/cgroup/blkio': No such file or directory
mount: 'pids'->'/sys/fs/cgroup/pids': No such file or directory
mount: 'schedtune'->'/sys/fs/cgroup/schedtune': No such file or directory
INFO[2023-01-11T05:16:52.029135861Z] Starting up
WARN[2023-01-11T05:16:52.038316123Z] could not change group /data/docker
/run/docker.sock to docker: group docker not found
INFO[2023-01-11T05:16:52.047296019Z] libcontainerd: started new containerd process
pid=2345
INFO[2023-01-11T05:16:52.047560603Z] parsed scheme: "unix"
module=grpc
INFO[2023-01-11T05:16:52.047635290Z] scheme "unix" not registered, fallback to
default scheme  module=grpc
INFO[2023-01-11T05:16:52.047750499Z] ccResolverWrapper: sending update to cc:
{[{unix:///data/docker/run/docker/containerd/containerd.sock  <nil> 0 <nil>}] <nil>
<nil>}  module=grpc
INFO[2023-01-11T05:16:52.047839665Z] ClientConn switching balancer to "pick_first"
module=grpc
WARN[0000] containerd config version `1` has been deprecated and will be removed in
containerd v2.0, please switch to version `2`, see https://github.com/containerd
/containerd/blob/main/docs/PLUGINS.md#version-header
INFO[2023-01-11T05:16:52.324794016Z] starting containerd
revision=9ba4b250366a5ddde94bb7c9d1def331423aa323.m version=v1.6.14.m
INFO[2023-01-11T05:16:53.373875082Z] loading plugin
"io.containerd.content.v1.content"...  type=io.containerd.content.v1
INFO[2023-01-11T05:16:53.381617583Z] loading plugin
"io.containerd.snapshotter.v1.aufs"...  type=io.containerd.snapshotter.v1
INFO[2023-01-11T05:16:53.397168731Z] skip loading plugin
"io.containerd.snapshotter.v1.aufs"...  error="aufs is not supported (modprobe aufs
failed: exit status 1 \"No module configuration directories given.\\nUsage:\\n\\n
modprobe [-alrqvsDb] [-d DIR] [MODULE]+\\n  modprobe [-alrqvsDb] [-d DIR] MODULE
[symbol=value][...]\\n\\nOptions:\\n  -b: Apply blocklist to module names too\\n
-d: Load modules from DIR, option may be used multiple times\\n  -D: Print
dependencies for modules only, do not load  -h: Print this help\\n  -l: List modules
matching pattern\\n  -r: Remove MODULE (multiple modules may be specified)\\n  -q:
Quiet\\n  -v: Verbose\\n\\n\"): skip plugin" type=io.containerd.snapshotter.v1
INFO[2023-01-11T05:16:53.397455970Z] loading plugin
"io.containerd.snapshotter.v1.devmapper"...  type=io.containerd.snapshotter.v1
WARN[2023-01-11T05:16:53.397576960Z] failed to load plugin
io.containerd.snapshotter.v1.devmapper  error="devmapper not configured"
INFO[2023-01-11T05:16:53.397659199Z] loading plugin
"io.containerd.snapshotter.v1.native"...  type=io.containerd.snapshotter.v1
INFO[2023-01-11T05:16:53.398052741Z] loading plugin
"io.containerd.snapshotter.v1.overlayfs"...  type=io.containerd.snapshotter.v1
INFO[2023-01-11T05:16:53.398761595Z] loading plugin
"io.containerd.snapshotter.v1.zfs"...  type=io.containerd.snapshotter.v1
INFO[2023-01-11T05:16:53.406010711Z] skip loading plugin
"io.containerd.snapshotter.v1.zfs"...  error="path /data/docker/lib/docker
```

```
/containerd/daemon/io.containerd.snapshotter.v1.zfs must be a zfs filesystem to be
used with the zfs snapshotter: skip plugin" type=io.containerd.snapshotter.v1
INFO[2023-01-11T05:16:53.406195034Z] loading plugin
"io.containerd.metadata.v1.bolt"...  type=io.containerd.metadata.v1
WARN[2023-01-11T05:16:53.406468940Z] could not use snapshotter devmapper in metadata
plugin  error="devmapper not configured"
INFO[2023-01-11T05:16:53.406561388Z] metadata content store policy set
policy=shared
INFO[2023-01-11T05:16:53.432757015Z] loading plugin
"io.containerd.differ.v1.walking"...  type=io.containerd.differ.v1
INFO[2023-01-11T05:16:53.432947484Z] loading plugin
"io.containerd.event.v1.exchange"...  type=io.containerd.event.v1
INFO[2023-01-11T05:16:53.433060661Z] loading plugin
"io.containerd.gc.v1.scheduler"...  type=io.containerd.gc.v1
INFO[2023-01-11T05:16:53.433261755Z] loading plugin
"io.containerd.service.v1.introspection-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.433409047Z] loading plugin
"io.containerd.service.v1.containers-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.433538422Z] loading plugin
"io.containerd.service.v1.content-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.434780141Z] loading plugin "io.containerd.service.v1.diff-
service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.434870297Z] loading plugin
"io.containerd.service.v1.images-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.434957641Z] loading plugin
"io.containerd.service.v1.leases-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.435038214Z] loading plugin
"io.containerd.service.v1.namespaces-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.435124828Z] loading plugin
"io.containerd.service.v1.snapshots-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.435321964Z] loading plugin
"io.containerd.runtime.v1.linux"...  type=io.containerd.runtime.v1
INFO[2023-01-11T05:16:53.435945818Z] loading plugin
"io.containerd.runtime.v2.task"...  type=io.containerd.runtime.v2
INFO[2023-01-11T05:16:53.436388109Z] loading plugin
"io.containerd.monitor.v1.cgroups"...  type=io.containerd.monitor.v1
INFO[2023-01-11T05:16:53.437673318Z] loading plugin "io.containerd.service.v1.tasks-
service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.437819308Z] loading plugin
"io.containerd.grpc.v1.introspection"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.437875141Z] loading plugin
"io.containerd.internal.v1.restart"...  type=io.containerd.internal.v1
INFO[2023-01-11T05:16:53.439832381Z] loading plugin
"io.containerd.grpc.v1.containers"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.439922329Z] loading plugin
"io.containerd.grpc.v1.content"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.439976912Z] loading plugin "io.containerd.grpc.v1.diff"...
type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440034881Z] loading plugin
"io.containerd.grpc.v1.events"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440090610Z] loading plugin
"io.containerd.grpc.v1.healthcheck"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440144204Z] loading plugin
"io.containerd.grpc.v1.images"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440202329Z] loading plugin
"io.containerd.grpc.v1.leases"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440254933Z] loading plugin
"io.containerd.grpc.v1.namespaces"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440333787Z] loading plugin
```

```
"io.containerd.internal.v1.opt"...  type=io.containerd.internal.v1
INFO[2023-01-11T05:16:53.440866964Z] loading plugin
"io.containerd.grpc.v1.snapshots"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440926600Z] loading plugin "io.containerd.grpc.v1.tasks"...
type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440981131Z] loading plugin
"io.containerd.grpc.v1.version"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.441035506Z] loading plugin
"io.containerd.tracing.processor.v1.otlp"...
type=io.containerd.tracing.processor.v1
INFO[2023-01-11T05:16:53.441108058Z] skip loading plugin
"io.containerd.tracing.processor.v1.otlp"...  error="no OpenTelemetry endpoint: skip
plugin" type=io.containerd.tracing.processor.v1
INFO[2023-01-11T05:16:53.441165818Z] loading plugin
"io.containerd.internal.v1.tracing"...  type=io.containerd.internal.v1
ERRO[2023-01-11T05:16:53.441272120Z] failed to initialize a tracing processor "otlp"
error="no OpenTelemetry endpoint: skip plugin"
INFO[2023-01-11T05:16:53.443261235Z] serving...
address=/data/docker/run/docker/containerd/containerd-debug.sock
INFO[2023-01-11T05:16:53.443461287Z] serving...
address=/data/docker/run/docker/containerd/containerd.sock.ttrpc
INFO[2023-01-11T05:16:53.443639464Z] serving...
address=/data/docker/run/docker/containerd/containerd.sock
INFO[2023-01-11T05:16:53.443709048Z] containerd successfully booted in 0.339054s
INFO[2023-01-11T05:16:53.467206081Z] parsed scheme: "unix"
module=grpc
INFO[2023-01-11T05:16:53.467304467Z] scheme "unix" not registered, fallback to
default scheme  module=grpc
INFO[2023-01-11T05:16:53.467370092Z] ccResolverWrapper: sending update to cc:
{[{unix:///data/docker/run/docker/containerd/containerd.sock  <nil> 0 <nil>}] <nil>
<nil>}  module=grpc
INFO[2023-01-11T05:16:53.467403738Z] ClientConn switching balancer to "pick_first"
module=grpc
INFO[2023-01-11T05:16:53.473255821Z] parsed scheme: "unix"
module=grpc
INFO[2023-01-11T05:16:53.473343686Z] scheme "unix" not registered, fallback to
default scheme  module=grpc
INFO[2023-01-11T05:16:53.473404467Z] ccResolverWrapper: sending update to cc:
{[{unix:///data/docker/run/docker/containerd/containerd.sock  <nil> 0 <nil>}] <nil>
<nil>}  module=grpc
INFO[2023-01-11T05:16:53.473436082Z] ClientConn switching balancer to "pick_first"
module=grpc
ERRO[2023-01-11T05:16:53.477520666Z] failed to mount overlay: no such device
storage-driver=overlay2
INFO[2023-01-11T05:16:53.486746812Z] stopping healthcheck following graceful
shutdown  module=libcontainerd
INFO[2023-01-11T05:16:53.486842698Z] stopping event stream following graceful
shutdown  error="context canceled" module=libcontainerd namespace=plugins.moby
failed to start daemon: error initializing graphdriver: driver not supported
```

I see some mount error at the beginning. Is that the issue?

I even tried method from **@daoudeddy** but no luck.

```
~ $ mkdir -p $PREFIX/var/service/dockerd/log
~ $ ln -sf $PREFIX/share/termux-services/svlogger $PREFIX/var/service/dockerd
/log/run
```

```
~ $ echo -e '#!/data/data/com.termux/files/usr/bin/sh\nexec sudo dockerd 2>&1' >
$PREFIX/var/service/dockerd/run
~ $  sv up dockerd
fail: dockerd: unable to change to service directory: file does not exist
```

Any idea how to fix this?

Thank you.

daoudeddy commented last week

> I have passed the Generally Necessary section but I still running into issue when I start the docker daemon. Anyone encounter the same issue?
>
> ```
> ~ $ sudo ./check-config.sh
> info: reading kernel config from /proc/config.gz ...
>
> Generally Necessary:
> - cgroup hierarchy: properly mounted [/dev]
> - CONFIG_NAMESPACES: enabled
> - CONFIG_NET_NS: enabled
> - CONFIG_PID_NS: enabled
> - CONFIG_IPC_NS: enabled
> - CONFIG_UTS_NS: enabled
> - CONFIG_CGROUPS: enabled
> - CONFIG_CGROUP_CPUACCT: enabled
> - CONFIG_CGROUP_DEVICE: enabled
> - CONFIG_CGROUP_FREEZER: enabled
> - CONFIG_CGROUP_SCHED: enabled
> - CONFIG_CPUSETS: enabled
> - CONFIG_MEMCG: enabled
> - CONFIG_KEYS: enabled
> - CONFIG_VETH: enabled
> - CONFIG_BRIDGE: enabled
> - CONFIG_BRIDGE_NETFILTER: enabled (as module)
> - CONFIG_IP_NF_FILTER: enabled
> - CONFIG_IP_NF_TARGET_MASQUERADE: enabled
> - CONFIG_NETFILTER_XT_MATCH_ADDRTYPE: enabled
> - CONFIG_NETFILTER_XT_MATCH_CONNTRACK: enabled
> - CONFIG_NETFILTER_XT_MATCH_IPVS: enabled
> - CONFIG_NETFILTER_XT_MARK: enabled
> - CONFIG_IP_NF_NAT: enabled
> - CONFIG_NF_NAT: enabled
> - CONFIG_POSIX_MQUEUE: enabled
> - CONFIG_DEVPTS_MULTIPLE_INSTANCES: enabled
> - CONFIG_NF_NAT_IPV4: enabled
> - CONFIG_NF_NAT_NEEDED: enabled
>
> Optional Features:
> - CONFIG_USER_NS: enabled
> - CONFIG_SECCOMP: enabled
> - CONFIG_SECCOMP_FILTER: enabled
> - CONFIG_CGROUP_PIDS: missing
> - CONFIG_MEMCG_SWAP: missing
> ```

```
- CONFIG_MEMCG_SWAP_ENABLED: missing
- CONFIG_MEMCG_KMEM: missing
- CONFIG_RESOURCE_COUNTERS: enabled
- CONFIG_IOSCHED_CFQ: enabled
- CONFIG_CFQ_GROUP_IOSCHED: missing
- CONFIG_BLK_CGROUP: missing
- CONFIG_BLK_DEV_THROTTLING: missing
- CONFIG_CGROUP_PERF: missing
- CONFIG_CGROUP_HUGETLB: missing
- CONFIG_NET_CLS_CGROUP: missing
- CONFIG_CGROUP_NET_PRIO: missing
- CONFIG_CFS_BANDWIDTH: missing
- CONFIG_FAIR_GROUP_SCHED: enabled
- CONFIG_RT_GROUP_SCHED: enabled
- CONFIG_IP_NF_TARGET_REDIRECT: enabled
- CONFIG_IP_VS: enabled
- CONFIG_IP_VS_NFCT: missing
- CONFIG_IP_VS_PROTO_TCP: missing
- CONFIG_IP_VS_PROTO_UDP: missing
- CONFIG_IP_VS_RR: missing
- CONFIG_SECURITY_SELINUX: enabled
- CONFIG_SECURITY_APPARMOR: missing
- CONFIG_EXT3_FS: enabled
- CONFIG_EXT3_FS_XATTR: enabled
- CONFIG_EXT3_FS_POSIX_ACL: missing
- CONFIG_EXT3_FS_SECURITY: missing
    (enable these ext3 configs if you are using ext3 as backing filesystem)
- CONFIG_EXT4_FS: enabled
- CONFIG_EXT4_FS_POSIX_ACL: missing
- CONFIG_EXT4_FS_SECURITY: enabled
    enable these ext4 configs if you are using ext4 as backing filesystem
- Network Drivers:
  - "overlay":
    - CONFIG_VXLAN: missing
    - CONFIG_BRIDGE_VLAN_FILTERING: missing
      Optional (for encrypted networks):
      - CONFIG_CRYPTO: enabled
      - CONFIG_CRYPTO_AEAD: enabled
      - CONFIG_CRYPTO_GCM: missing
      - CONFIG_CRYPTO_SEQIV: enabled
      - CONFIG_CRYPTO_GHASH: missing
      - CONFIG_XFRM: enabled
      - CONFIG_XFRM_USER: enabled
      - CONFIG_XFRM_ALGO: enabled
      - CONFIG_INET_ESP: enabled
      - CONFIG_INET_XFRM_MODE_TRANSPORT: enabled
  - "ipvlan":
    - CONFIG_IPVLAN: missing
  - "macvlan":
    - CONFIG_MACVLAN: missing
    - CONFIG_DUMMY: enabled
  - "ftp,tftp client in container":
    - CONFIG_NF_NAT_FTP: enabled
    - CONFIG_NF_CONNTRACK_FTP: enabled
    - CONFIG_NF_NAT_TFTP: enabled
    - CONFIG_NF_CONNTRACK_TFTP: enabled
- Storage Drivers:
  - "aufs":
    - CONFIG_AUFS_FS: missing
```

```
    - "btrfs":
      - CONFIG_BTRFS_FS: missing
      - CONFIG_BTRFS_FS_POSIX_ACL: missing
    - "devicemapper":
      - CONFIG_BLK_DEV_DM: enabled
      - CONFIG_DM_THIN_PROVISIONING: missing
    - "overlay":
      - CONFIG_OVERLAY_FS: missing
    - "zfs":
      - /dev/zfs: missing
      - zfs command: missing
      - zpool command: missing

  Limits:
  - /proc/sys/kernel/keys/root_maxkeys: 1000000
```

Here is the error when I run the command:

```
~ $ sudo dockerd --iptables=false
mount: 'cgroup2_root'->'/sys/fs/cgroup/cg2_bpf': No such device
mount: 'blkio'->'/sys/fs/cgroup/blkio': No such file or directory
mount: 'pids'->'/sys/fs/cgroup/pids': No such file or directory
mount: 'schedtune'->'/sys/fs/cgroup/schedtune': No such file or directory
INFO[2023-01-11T05:16:52.029135861Z] Starting up
WARN[2023-01-11T05:16:52.038316123Z] could not change group /data/docker
/run/docker.sock to docker: group docker not found
INFO[2023-01-11T05:16:52.047296019Z] libcontainerd: started new containerd
process  pid=2345
INFO[2023-01-11T05:16:52.047560603Z] parsed scheme: "unix"
module=grpc
INFO[2023-01-11T05:16:52.047635290Z] scheme "unix" not registered, fallback to
default scheme  module=grpc
INFO[2023-01-11T05:16:52.047750499Z] ccResolverWrapper: sending update to cc:
{[{unix:///data/docker/run/docker/containerd/containerd.sock  <nil> 0 <nil>}]
<nil> <nil>}  module=grpc
INFO[2023-01-11T05:16:52.047839665Z] ClientConn switching balancer to
"pick_first"  module=grpc
WARN[0000] containerd config version `1` has been deprecated and will be
removed in containerd v2.0, please switch to version `2`, see
https://github.com/containerd/containerd/blob/main/docs/PLUGINS.md#version-
header
INFO[2023-01-11T05:16:52.324794016Z] starting containerd
revision=9ba4b250366a5ddde94bb7c9d1def331423aa323.m version=v1.6.14.m
INFO[2023-01-11T05:16:53.373875082Z] loading plugin
"io.containerd.content.v1.content"...  type=io.containerd.content.v1
INFO[2023-01-11T05:16:53.381617583Z] loading plugin
"io.containerd.snapshotter.v1.aufs"...  type=io.containerd.snapshotter.v1
INFO[2023-01-11T05:16:53.397168731Z] skip loading plugin
"io.containerd.snapshotter.v1.aufs"...  error="aufs is not supported (modprobe
aufs failed: exit status 1 \"No module configuration directories
given.\\nUsage:\\n\\n  modprobe [-alrqvsDb] [-d DIR] [MODULE]+\\n  modprobe
[-alrqvsDb] [-d DIR] MODULE [symbol=value][...]\\n\\nOptions:\\n  -b: Apply
blocklist to module names too\\n  -d: Load modules from DIR, option may be used
multiple times\\n  -D: Print dependencies for modules only, do not load  -h:
Print this help\\n  -l: List modules matching pattern\\n  -r: Remove MODULE
(multiple modules may be specified)\\n  -q: Quiet\\n  -v: Verbose\\n\\n\"):
skip plugin" type=io.containerd.snapshotter.v1
```

```
INFO[2023-01-11T05:16:53.397455970Z] loading plugin
"io.containerd.snapshotter.v1.devmapper"...  type=io.containerd.snapshotter.v1
WARN[2023-01-11T05:16:53.397576960Z] failed to load plugin
io.containerd.snapshotter.v1.devmapper  error="devmapper not configured"
INFO[2023-01-11T05:16:53.397659199Z] loading plugin
"io.containerd.snapshotter.v1.native"...  type=io.containerd.snapshotter.v1
INFO[2023-01-11T05:16:53.398052741Z] loading plugin
"io.containerd.snapshotter.v1.overlayfs"...  type=io.containerd.snapshotter.v1
INFO[2023-01-11T05:16:53.398761595Z] loading plugin
"io.containerd.snapshotter.v1.zfs"...  type=io.containerd.snapshotter.v1
INFO[2023-01-11T05:16:53.406010711Z] skip loading plugin
"io.containerd.snapshotter.v1.zfs"...  error="path /data/docker/lib/docker
/containerd/daemon/io.containerd.snapshotter.v1.zfs must be a zfs filesystem to
be used with the zfs snapshotter: skip plugin"
type=io.containerd.snapshotter.v1
INFO[2023-01-11T05:16:53.406195034Z] loading plugin
"io.containerd.metadata.v1.bolt"...  type=io.containerd.metadata.v1
WARN[2023-01-11T05:16:53.406468940Z] could not use snapshotter devmapper in
metadata plugin  error="devmapper not configured"
INFO[2023-01-11T05:16:53.406561388Z] metadata content store policy set
policy=shared
INFO[2023-01-11T05:16:53.432757015Z] loading plugin
"io.containerd.differ.v1.walking"...  type=io.containerd.differ.v1
INFO[2023-01-11T05:16:53.432947484Z] loading plugin
"io.containerd.event.v1.exchange"...  type=io.containerd.event.v1
INFO[2023-01-11T05:16:53.433060661Z] loading plugin
"io.containerd.gc.v1.scheduler"...  type=io.containerd.gc.v1
INFO[2023-01-11T05:16:53.433261755Z] loading plugin
"io.containerd.service.v1.introspection-service"...
type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.433409047Z] loading plugin
"io.containerd.service.v1.containers-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.433538422Z] loading plugin
"io.containerd.service.v1.content-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.434780141Z] loading plugin
"io.containerd.service.v1.diff-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.434870297Z] loading plugin
"io.containerd.service.v1.images-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.434957641Z] loading plugin
"io.containerd.service.v1.leases-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.435038214Z] loading plugin
"io.containerd.service.v1.namespaces-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.435124828Z] loading plugin
"io.containerd.service.v1.snapshots-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.435321964Z] loading plugin
"io.containerd.runtime.v1.linux"...  type=io.containerd.runtime.v1
INFO[2023-01-11T05:16:53.435945818Z] loading plugin
"io.containerd.runtime.v2.task"...  type=io.containerd.runtime.v2
INFO[2023-01-11T05:16:53.436388109Z] loading plugin
"io.containerd.monitor.v1.cgroups"...  type=io.containerd.monitor.v1
INFO[2023-01-11T05:16:53.437673318Z] loading plugin
"io.containerd.service.v1.tasks-service"...  type=io.containerd.service.v1
INFO[2023-01-11T05:16:53.437819308Z] loading plugin
"io.containerd.grpc.v1.introspection"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.437875141Z] loading plugin
"io.containerd.internal.v1.restart"...  type=io.containerd.internal.v1
INFO[2023-01-11T05:16:53.439832381Z] loading plugin
"io.containerd.grpc.v1.containers"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.439922329Z] loading plugin
```

```
"io.containerd.grpc.v1.content"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.439976912Z] loading plugin
"io.containerd.grpc.v1.diff"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440034881Z] loading plugin
"io.containerd.grpc.v1.events"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440090610Z] loading plugin
"io.containerd.grpc.v1.healthcheck"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440144204Z] loading plugin
"io.containerd.grpc.v1.images"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440202329Z] loading plugin
"io.containerd.grpc.v1.leases"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440254933Z] loading plugin
"io.containerd.grpc.v1.namespaces"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440333787Z] loading plugin
"io.containerd.internal.v1.opt"...  type=io.containerd.internal.v1
INFO[2023-01-11T05:16:53.440866964Z] loading plugin
"io.containerd.grpc.v1.snapshots"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440926600Z] loading plugin
"io.containerd.grpc.v1.tasks"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.440981131Z] loading plugin
"io.containerd.grpc.v1.version"...  type=io.containerd.grpc.v1
INFO[2023-01-11T05:16:53.441035506Z] loading plugin
"io.containerd.tracing.processor.v1.otlp"...
type=io.containerd.tracing.processor.v1
INFO[2023-01-11T05:16:53.441108058Z] skip loading plugin
"io.containerd.tracing.processor.v1.otlp"...  error="no OpenTelemetry endpoint:
skip plugin" type=io.containerd.tracing.processor.v1
INFO[2023-01-11T05:16:53.441165818Z] loading plugin
"io.containerd.internal.v1.tracing"...  type=io.containerd.internal.v1
ERRO[2023-01-11T05:16:53.441272120Z] failed to initialize a tracing processor
"otlp"  error="no OpenTelemetry endpoint: skip plugin"
INFO[2023-01-11T05:16:53.443261235Z] serving...
address=/data/docker/run/docker/containerd/containerd-debug.sock
INFO[2023-01-11T05:16:53.443461287Z] serving...
address=/data/docker/run/docker/containerd/containerd.sock.ttrpc
INFO[2023-01-11T05:16:53.443639464Z] serving...
address=/data/docker/run/docker/containerd/containerd.sock
INFO[2023-01-11T05:16:53.443709048Z] containerd successfully booted in
0.339054s
INFO[2023-01-11T05:16:53.467206081Z] parsed scheme: "unix"
module=grpc
INFO[2023-01-11T05:16:53.467304467Z] scheme "unix" not registered, fallback to
default scheme  module=grpc
INFO[2023-01-11T05:16:53.467370092Z] ccResolverWrapper: sending update to cc:
{[{unix:///data/docker/run/docker/containerd/containerd.sock  <nil> 0 <nil>}]
<nil> <nil>}  module=grpc
INFO[2023-01-11T05:16:53.467403738Z] ClientConn switching balancer to
"pick_first"  module=grpc
INFO[2023-01-11T05:16:53.473255821Z] parsed scheme: "unix"
module=grpc
INFO[2023-01-11T05:16:53.473343686Z] scheme "unix" not registered, fallback to
default scheme  module=grpc
INFO[2023-01-11T05:16:53.473404467Z] ccResolverWrapper: sending update to cc:
{[{unix:///data/docker/run/docker/containerd/containerd.sock  <nil> 0 <nil>}]
<nil> <nil>}  module=grpc
INFO[2023-01-11T05:16:53.473436082Z] ClientConn switching balancer to
"pick_first"  module=grpc
ERRO[2023-01-11T05:16:53.477520666Z] failed to mount overlay: no such device
storage-driver=overlay2
```

> ```
> INFO[2023-01-11T05:16:53.486746812Z] stopping healthcheck following graceful
> shutdown  module=libcontainerd
> INFO[2023-01-11T05:16:53.486842698Z] stopping event stream following graceful
> shutdown  error="context canceled" module=libcontainerd namespace=plugins.moby
> failed to start daemon: error initializing graphdriver: driver not supported
> ```
>
> I see some mount error at the beginning. Is that the issue?
>
> I even tried method from **@daoudeddy** but no luck.
>
> ```
> ~ $ mkdir -p $PREFIX/var/service/dockerd/log
> ~ $ ln -sf $PREFIX/share/termux-services/svlogger $PREFIX/var/service/dockerd
> /log/run
> ~ $ echo -e '#!/data/data/com.termux/files/usr/bin/sh\nexec sudo dockerd 2>&1'
> > $PREFIX/var/service/dockerd/run
> ~ $  sv up dockerd
> fail: dockerd: unable to change to service directory: file does not exist
> ```
>
> Any idea how to fix this?
>
> Thank you.

You missed these commands

```
sudo mount -t tmpfs -o mode=755 tmpfs /sys/fs/cgroup
sudo mkdir -p /sys/fs/cgroup/devices
sudo mount -t cgroup -o devices cgroup /sys/fs/cgroup/devices
```

owen31302 commented last week

Hi **@daoudeddy** ,

Thanks for your prompt reply. I ran the above commands but still got the same error.

```
mount: 'cgroup2_root'->'/sys/fs/cgroup/cg2_bpf': No such device
mount: 'blkio'->'/sys/fs/cgroup/blkio': No such file or directory
mount: 'pids'->'/sys/fs/cgroup/pids': No such file or directory
mount: 'schedtune'->'/sys/fs/cgroup/schedtune': No such file or directory
...
```

I will do some research about mounting and update this post if I got this working.

daoudeddy commented last week • edited

> Hi **@daoudeddy** ,
>
> Thanks for your prompt reply. I ran the above commands but still got the same error.

> ```
> mount: 'cgroup2_root'->'/sys/fs/cgroup/cg2_bpf': No such device
> mount: 'blkio'->'/sys/fs/cgroup/blkio': No such file or directory
> mount: 'pids'->'/sys/fs/cgroup/pids': No such file or directory
> mount: 'schedtune'->'/sys/fs/cgroup/schedtune': No such file or directory
> ...
> ```
>
> I will do some research about mounting and update this post if I got this working.

@**owen31302** can you run `ls -la` in /sys/fs/cgroup and post the result?

owen31302 commented 5 days ago • edited

> Hi @**daoudeddy** ,
> Thanks for your prompt reply. I ran the above commands but still got the same error.
>
> ```
> mount: 'cgroup2_root'->'/sys/fs/cgroup/cg2_bpf': No such device
> mount: 'blkio'->'/sys/fs/cgroup/blkio': No such file or directory
> mount: 'pids'->'/sys/fs/cgroup/pids': No such file or directory
> mount: 'schedtune'->'/sys/fs/cgroup/schedtune': No such file or directory
> ...
> ```
>
> I will do some research about mounting and update this post if I got this working.
>
> @**owen31302** can you run `ls -la` in /sys/fs/cgroup and post the result?

@**daoudeddy** Here is the result

```
~ $ ls -la /sys/fs/cgroup
total 0
drwxr-xr-x  9 root   root   180 Jan 12 03:52 .
drwxr-xr-x  9 root   root     0 Jul  3  1970 ..
drwxr-xr-x  2 root   root    40 Jan 11 23:49 cg2_bpf
dr-xr-xr-x  2 system system   0 Jul  3  1970 cpu
dr-xr-xr-x 99 root   root     0 Jul  3  1970 cpuacct
dr-xr-xr-x  8 system system   0 Jul  3  1970 cpuset
dr-xr-xr-x  2 root   root     0 Jan 11 23:39 devices
dr-xr-xr-x  2 root   root     0 Jan 11 23:39 freezer
dr-xr-xr-x  4 root   root     0 Jul  3  1970 memory
```

I also checked the mounting

```
~ $ cat /proc/mounts | grep cgroup
none /dev/cpuctl cgroup rw,nosuid,nodev,noexec,relatime,cpu 0 0
none /dev/cpuset cgroup
rw,nosuid,nodev,noexec,relatime,cpuset,noprefix,release_agent=/sbin
/cpuset_release_agent 0 0
none /dev/memcg cgroup rw,nosuid,nodev,noexec,relatime,memory 0 0
none /acct cgroup rw,nosuid,nodev,noexec,relatime,cpuacct 0 0
```

```
cgroup_root /sys/fs/cgroup tmpfs rw,seclabel,nosuid,nodev,noexec,relatime 0 0
cpu /sys/fs/cgroup/cpu cgroup rw,nosuid,nodev,noexec,relatime,cpu 0 0
cpuacct /sys/fs/cgroup/cpuacct cgroup rw,nosuid,nodev,noexec,relatime,cpuacct 0 0
cpuset /sys/fs/cgroup/cpuset cgroup
rw,nosuid,nodev,noexec,relatime,cpuset,noprefix,release_agent=/sbin
/cpuset_release_agent 0 0
devices /sys/fs/cgroup/devices cgroup rw,nosuid,nodev,noexec,relatime,devices 0 0
freezer /sys/fs/cgroup/freezer cgroup rw,nosuid,nodev,noexec,relatime,freezer 0 0
memory /sys/fs/cgroup/memory cgroup rw,nosuid,nodev,noexec,relatime,memory 0 0
tmpfs /sys/fs/cgroup tmpfs rw,seclabel,relatime,mode=755 0 0
cgroup /sys/fs/cgroup/devices cgroup rw,relatime,devices 0 0
cpu /sys/fs/cgroup/cpu cgroup rw,nosuid,nodev,noexec,relatime,cpu 0 0
cpuacct /sys/fs/cgroup/cpuacct cgroup rw,nosuid,nodev,noexec,relatime,cpuacct 0 0
cpuset /sys/fs/cgroup/cpuset cgroup
rw,nosuid,nodev,noexec,relatime,cpuset,noprefix,release_agent=/sbin
/cpuset_release_agent 0 0
freezer /sys/fs/cgroup/freezer cgroup rw,nosuid,nodev,noexec,relatime,freezer 0 0
memory /sys/fs/cgroup/memory cgroup rw,nosuid,nodev,noexec,relatime,memory 0 0
```

Or maybe I should fix the error

```
ERRO[2023-01-11T05:16:53.477520666Z] failed to mount overlay: no such device
storage-driver=overlay2
```

by enabling

```
CONFIG_OVERLAY_FS
```

Still doing the research about how to fix the other error

```
ERRO[2023-01-11T05:16:53.441272120Z] failed to initialize a tracing processor "otlp"
error="no OpenTelemetry endpoint: skip plugin"
```

Edit:
I am able to start docker after several try and here are what I have done:

1. Enabling CONFIG_OVERLAY_FS <= does not work
2. Did the following command <= does not work

```
mount -t tmpfs -o mode=755 tmpfs /sys/fs/cgroup
mkdir -p /sys/fs/cgroup/devices
mount -t cgroup -o devices cgroup /sys/fs/cgroup/devices
```

3. Change overlay2 to overlay works for me

```
~ $ sudo cat $PREFIX/etc/docker/daemon.json
{
    "data-root": "/data/docker/lib/docker",
    "exec-root": "/data/docker/run/docker",
```

```
    "pidfile": "/data/docker/run/docker.pid",
    "hosts": [
        "unix:///data/docker/run/docker.sock"
    ],
    "storage-driver": "overlay"
}
```

https://gist.github.com/FreddieOliveira/efe850df7ff3951cb62d74bd770dce27?permalink_comment_id=4126099#gistcomment-4126099

I guess the combination of step 2 and step 3 helps me.