University of Brighton

Project Analysis

Adam Pietrzycki

December 8, 2016

Abstract

In this document I will be keeping a log of my approach and analysis of this project, note that not everything in this will be of any use.

Contents

1 Original pi-gen **2**
1.1 Main build files 2
1.1.1 config 2
1.1.2 SCRIPTS/common 2
1.1.3 SCRIPTS/dependencies_check 3
1.1.4 build.sh 3

2 NB **4**
2.1 Kernel panic when Virtualizing 4

Chapter 1

Original pi-gen

1.1 Main build files

The build.sh file is the one you run to start generating the Raspbian images. It first sets up a few EXPORTS and SOURCES files from the scripts folder.

1.1.1 config

In the config file you can set an IMG_NAME and an APT_PROXY. The default file can just contain "IMG_NAME='Raspbian'". A quick thing you can do to set up this file is:

```
echo "IMG_NAME='Raspbian'" > config

NOTE:
>   is overwrite if present, create if not.
>> is add to end of file it present, create if not.
```

1.1.2 SCRIPTS/common

log

Gets current time and uses a pipe with *tee* to write to the log file.
(`http://man7.org/linux/man-pages/man1/tee.1.html`)

bootstrap

Sets up debootstrap, uses *capsh* to create env I think?
(`http://man7.org/linux/man-pages/man1/capsh.1.html`)

copy_previous

If rootfs folder doesn't exist it will create one, if it does then it uses *rsync* to copy from previous to current stage. This can be avoided to speed things up?
(`http://linuxcommand.org/man_pages/rsync1.html`)

unmount

Does a few checks using $1, unmouts mounted folders using *umount*.
(`http://man7.org/linux/man-pages/man8/umount.8.html`)

unmount_image

First syncs then get *losetup*, then it does a loop through the directories and uses *unmount()*, finally *kpartx* and then *losetup* again.

- `http://linuxcommand.org/man_pages/losetup8.html`
- `http://www.dsm.fordham.edu/cgi-bin/man-cgi.pl?topic=kpartx&ampsect=8`

onchroot

Mounts with bind, uses *realpath and capsh again*:

- $ROOTFS_DIR/proc
- $ROOTFS_DIR/dev
- $ROOTFS_DIR/dev/pts
- $ROOTFS_DIR/sys

(`http://man7.org/linux/man-pages/man3/realpath.3.html`)

update_issue

Prints pi-gen version? This is strange; look into it but it does not look like a priority.

1.1.3 SCRIPTS/dependencies_check

Dependencies_check, checks if each required tool is installed on the system, the list of packages required can be found in the root directory in ../DEPENDS file.

1.1.4 build.sh

Exports and Source

Stage

Sub-Stage

log

Chapter 2

NB

2.1 Kernel panic when Virtualizing

Different versions of pi-gen would fail at different times, normally in stage 4. This was due to code being reverted thus removing the setting a max build stage functionality. The bento boxes are the safest to use though need to install a few more packages, they come with a 50GB virtual disk where as other Vagrant images came with the standard of 8GB and it was a pain to increase.