Building a Debian Root filesystem

Zilogic Systems

1. Manually built root filesystem

- BusyBox is used to build the various shell commands and utilities.
- Other applications are built manually by fetching it's source code.

Pros

- Small sized root filesystem.
- Flexibility in selecting applications and their versions.

Cons

- Utilities available in Busybox is limited.
- Building applications manually from source is tedious.
- Since lot of manual work is involved, replicating the build process is difficult

2. Pre-built Root filesystem

- Root filesystem is built from existing binaries.
- Building root filesystem involves just installing the binaries.

Pros

- Building the filesystem is simple and fast.

Cons

- Choosing application version is limited as the pre-built version is fixed.
- Pre-built binaries are not available for all architectures.

3. Debian on ARM

- We will build an ARM based Debian root filesystem.
- A pendrive containing the Debian root filesystem, will then be used for booting the target.
- Multistrap is a tool to create root filesystems from pre-built applications.

3.1. Steps to Build Rootfs using Multistrap

- Multistrap requires a configuration file
- Configuration file contains the list of packages to be installed and the source from which the packages are downloaded.

  multistrap.conf.

```
[General]
noauth=true
debootstrap=Packages
aptsources=Packages

[Packages]
```

```
packages=bash base-files base-passwd e2fsprogs login mount sysvinit
util-linux diffutils findutils gawk grep sed gzip apt netbase hostname
source=http://ftp.us.debian.org/debian
suite=wheezy
```

- The following command builds the root filesystem.

```
host$ /usr/sbin/multistrap -a armel -d rootfs -f multistrap.conf
```

- `-a` specify the architecture
- `-f` location of the configuration file
- `-d` Root filesystem path
- Copy the root filesystem to a pendrive.

```
host$ cp -a /path/to/rootfs/* /path/to/pendrive/
```

- Boot the root filesystem with `init` boot argument set to `/bin/sh`.

```
console=ttyS0,115200 root=/dev/sda1 init=/bin/sh rootwait
```

- The packages in the root filesystem needs to be configured
- Execute the following commands in the target

```
target# chown 0:0 -R /bin /usr/bin /sbin /usr/sbin
target# mount -t proc nodev /proc
target# export PATH=/usr/sbin:/usr/bin:/sbin:/bin
target# /var/lib/dpkg/info/dash.preinst install
target# DEBIAN_FRONTEND=noninteractive dpkg --configure -a
```

- Change the root password using `passwd` command.

```
target# passwd
```

- The following files needs to be modified.
  - `/etc/inittab`
  - `/etc/hostname`
  - `/etc/fstab`
- Uncomment the following line in `/etc/inittab` and update the baudrate. This will cause login prompt to be displayed on the serial port.

```
T0:23:respawn:/sbin/getty -L ttyS0 115200 vt100
```

- Add the following lines in `/etc/fstab` to automount /proc.

```
proc /proc proc defaults 0 0
```

- Create the file `/etc/hostname` with the name of the host.
- Reboot and remove init argument from bootargs.

4. Further Reading

- `man` page for `multistrap`
- Building a small Debian root filesystem with Multistrap [http://free-electrons.com/blog/embdebian-with-multistrap/]