The State of PTXdist

Embedded Linux Conference Europe 2020

Roland Hieber <rhi@pengutronix.de>
What is PTXdist?

• Build system with focus on building Embedded Linux images from source

• Building blocks:
  – GNU Make
  – Kconfig
  – Bash
  – AWK

• Monthly releases
  – First version from before August 2003

• GPL-2.0 licensed

• https://www.ptxdist.org
A Typical BSP

Philosophy

- BSP structure == PTXdist structure
- Overwrite and extend upstream files locally
A First Look

$ ptxdist menuconfig

$ ptxdist platformconfig
Applying Patches

- Automatically applied during the extract stage
- Edit patch queue with *git* or *quilt*
Package Types

Target packages
• Built on the build host for the target architecture
  – e.g: systemd, busybox, coreutils, kernel, bootloader

Host packages
• Built on the host, executed on the host
• Compatible build environment on different host systems
  – e.g.: host-dosfstools, host-python3, host-cmake

Image packages
• Determine the image format and the list of installed packages
  – e.g.: hdimage, root.tgz, RAUC bundles
Creating New Package Rules

~/projects/my-bsp $ ptxdist newpackage target

ptxdist: creating a new 'target' package:

ptxdist: enter package name............: mypackage
ptxdist: enter version number.........: 0.1
ptxdist: enter URL of basedir.........: https://example.org/mypackage
ptxdist: enter suffix.................: tar.gz
ptxdist: enter package author.........: Roland Hieber <rhi@pengutronix.de>
ptxdist: enter package section.........: project_specific
ptxdist: select option by number:
ptxdist: [1] autoconf
ptxdist: [2] cmake
ptxdist: [3] kconfig
ptxdist: [4] meson
ptxdist: [5] perl
ptxdist: [6] python3
ptxdist: [7] qmake
ptxdist: conf tool....................: 2

generating rules/mypackage.make
generating rules/mypackage.in
Package Definition: rules/mypackage.in

```plaintext
## SECTION=project_specific

config MYPACKAGE
  tristate
  select HOST_CMAKE
  select LIBUSB
  prompt "mypackage"
  help
    MyPackage is an example package.
    It is built with CMake and uses libusb at runtime.
```
Package Definition: rules/mypackage.make

PACKAGES-$(PTXCONF_MYPACKAGE) += mypackage

MYPACKAGE_VERSION := 0.1
MYPACKAGE_MD5 := 68b329da9893e34099c7d8ad5cb9c940
MYPACKAGE := mypackage-$(MYPACKAGE_VERSION)
MYPACKAGE_SUFFIX := tar.gz
MYPACKAGE_URL := https://ftp.example.org/mypackage/$(MYPACKAGE).$(MYPACKAGE_SUFFIX)
MYPACKAGE_SOURCE := $(SRCDIR)/$(MYPACKAGE).$(MYPACKAGE_SUFFIX)
MYPACKAGE_DIR := $(BUILDDIR)/$(MYPACKAGE)
MYPACKAGE_LICENSE := 0BSD
MYPACKAGE_LICENSE_FILES := file://LICENSE;md5=60b725f10c9c85c70d97880dfe8191b3
Package Definition: rules/mypackage.make

# Prepare
#
#
MYPACKAGE_CONF_TOOL := cmake
MYPACKAGE_CONF_OPT := $(CROSS_CMAKE_USR) -DUSB=ON
#
#
# Target-Install
#
$(STATEDIR)/mypackage.targetinstall:
  @$call targetinfo

  @$call install_init, mypackage
  @$call install_fixup, mypackage,PRIORITY,optional
  @$call install_fixup, mypackage,SECTION,base
  @$call install_fixup, mypackage,AUTHOR,"Roland Hieber <rhi@pengutronix.de>"
  @$call install_fixup, mypackage,DESCRIPTION,missing

  @$call install_copy, mypackage, 0, 0, 0755, -, /usr/bin/myprog

  @$call install_finish, mypackage

  @$call touch
Layer Mechanisms

• Alternative config files
  – Customising config files

• Platforms
  – Different hardware

• Collections
  – Different sets of software packages

• Kconfig diffs
  – Similar kernel/bootloader packages

• Base layers
  – Free-style layering
Alternative Config Files

```bash
# ptxdist/rules/cups.make
$(STATEDIR)/cups.targetinstall:
    # ...
    @$({call} install_alternative, cups, daemon, lp, 0640, /etc/printcap)
```

Search order (roughly):

1. my-bsp/projectroot/etc/printcap
2. my-bsp/configs/platform-v7a/projectroot/etc/printcap
3. ptxdist/projectroot/etc/printcap
4. /etc/printcap from `cups` install dir
5. /etc/printcap from `cups` build dir
Platforms vs. Userland

userland
- systemd
- busybox
- cups

platform-rpi
arm-1136jfs-linux-gnueabihf
- kernel
  - (4.19) kernel version
- barebox
- image_hdimg
- image_boot_vfat
# Platforms vs. Userland

- **userland**
  - systemd
  - busybox
  - cups

---

- **platform-rpi**
  - arm-1136jfs-linux-gnueabihf
  - kernel (4.19) kernel version
  - barebox
  - image_hdimg
  - image_boot_vfat

- **platform-v7a**
  - arm-v7a-linux-gnueabihf
  - kernel (5.8.9) kernel version
  - barebox
  - image_hdimg
  - image_boot_vfat

- **platform-v8a**
  - aarch64-v8a-linux-gnu
  - kernel (5.9-rc8) kernel version
  - barebox
  - image_hdimg
  - image_boot_vfat
Collections

**debug collection**
- gdb
- strace
- stress-ng

**release collection**
- gdb
- strace
- stress-ng

**userland**
- systemd
- busybox
- cups

**platform-rpi**
- arm-1136jfs-linux-gnueabihf
  - kernel (4.19): kernel version
  - barebox
  - image_hdimg
  - image_boot_vfat

**platform-v7a**
- arm-v7a-linux-gnueabihf
  - kernel (5.8.9): kernel version
  - barebox
  - image_hdimg
  - image_boot_vfat

**platform-v8a**
- aarch64-v8a-linux-gnu
  - kernel (5.9-rc8): kernel version
  - barebox
  - image_hdimg
  - image_boot_vfat
Collections: Scenarios

1) Collection used by image

```
IMAGE_DEBUG_TGZ := image-debug-tgz
IMAGE_DEBUG_TGZ_DIR := $(BUILDDIR)/$(IMAGE_DEBUG_TGZ)
IMAGE_DEBUG_TGZ_IMAGE := $(IMAGEDIR)/debug.tgz
IMAGE_DEBUG_TGZ_PKGS = $(call ptx/collection, $(call ptx/in-path, \ 
PTXDIST_PATH_LAYERS, configs/debug.collection))
```

2) Collection selected by user

- All images will use this collection (unless scenario 1.)

```
~/projects/my-bsp $ ptxdist collection configs/debug.collection
info: selected collection config: 'configs/debug.collection'

~/projects/my-bsp $ ptxdist -q -j -k images
```
Kconfig Diffs

# configs/platform-v7a/rules/kernel-debug.make
PACKAGES-$(PTXCONF_KERNEL_DEBUG) += kernel-debug
KERNEL_DEBUG_VERSION := 5.4
# ...
KERNEL_DEBUG_CONFIG := $(call ptx/in-platformconfigdir, kernelconfig-debug)
KERNEL_DEBUG_REF_CONFIG := $(call ptx/in-platformconfigdir, kernelconfig)

~/projects/my-bsp $ ptxdist menuconfig kernel-debug
# [... enable debug options ... ]
~/projects/my-bsp $ ls -l configs/platform-v7a/kernelconfig*
configs/platform-v7a/kernelconfig
configs/platform-v7a/kernelconfig-debug
configs/platform-v7a/kernelconfig-debug.diff

~/projects/my-bsp $ cat configs/platform-v7a/kernelconfig-debug.diff
1cdfdb2da785381a41fdd7320b37cd3d
CONFIG_DMA_API_DEBUG=y
CONFIG_DMA_API_DEBUG_SG=y

MD5 sum of ref config
Base Layers

distrokit/
  configs
    platform-v7a
      kernelconfig
      platformconfig
    ptxconfig
  rules
    datapartition.in
    datapartition.make

my-bsp/
  base -> ../distrokit/
  configs
    ptxconfig
    ptxconfig.diff
  rules
    mypackage.in
    mypackage.make

Base layer reference
Kconfig diff
Base Layers: Caveats

~/.projects/my-bsp $ cat configs/ptxconfig.diff
105d4dcf6b0783e7fe428f27eac0f43e
PTXCONF_VIM=y

• Layer dependencies are static
  – Kconfig diffs are pinned down via MD5
  – ptxdist oldconfig synchronises diffs
  – Solve update conflicts early

• PTXdist is always implicitly the lowest layer
Code Signing Infrastructure

Use Cases

• Verified Boot / High Assurance Boot

  ROM code → bootloader → kernel FIT → file system

• Signed update bundles

Different signature providers

• Development phase (e.g. SoftHSM)
• Release phase (e.g. NitroKey HSM, Cloud service)
Code Signing Infrastructure

Providers

• Handles communication with HSM
• Selected in platformconfig menu
  – e.g. host-devel-code-signing, host-release-code-signing
• ptxdist newpackage code-signing-provider

Consumers

• Package recipes
  – e.g. barebox, kernel-fit, image-rauc
```bash
~/projects/my-bsp $ ptxdist bsp-info

---------------------
target: bsp-info
---------------------

vendor: ACME
project: MyBSP
version: -

platform: v7a
platform version: -

BSP: /home/rohieb/projects/my-bsp
PTXdist: /usr/local/lib/ptxdist-2020.01.0

ptxconfig: my-bsp/configs/ptxconfig
platformconfig: my-bsp/configs/platform-v7a/platformconfig

images: image-hdim
        image-kernel
        image-root-ext
        image-root-tgz

finished target bsp-info
```
```bash
~/.projects/my-bsp $ ptxdist package-info busybox

target: busybox.package-info

package: busybox
version: 1.31.1
license: GPL-2.0-only
files: file://LICENSE;md5=de10de48642ab74318e893a61105afbb
source: /usr/src/busybox-1.31.1.tar.bz2
md5: 70913edaf2263a157393af07565c17f0
url: https://www.busybox.net/downloads/busybox-1.31.1.tar.bz2
src dir: my-bsp/platform-v7a/build-target/busybox-1.31.1
build dir: my-bsp/platform-v7a/build-target/busybox-1.31.1
pkg dir: my-bsp/platform-v7a/packages/busybox-1.31.1
rule file: ptxdist-2020.09.0/rules/busybox.make
menu file: ptxdist-2020.09.0/rules/busybox.in
patches: ptxdist-2020.09.0/patches/busybox-1.31.1

finished target busybox.package-info
```
License Report

PACKAGES-$(PTXCONF_MYPACKAGE) += mypackage

MYPACKAGE_VERSION := 0.1
MYPACKAGE_MD5 := 68b329da9893e34099c7d8ad5cb9c940
MYPACKAGE := mypackage-$(MYPACKAGE_VERSION)
MYPACKAGE_SUFFIX := tar.gz
MYPACKAGE_URL := https://ftp.example.org/mypackage/$(MYPACKAGE).$(MYPACKAGE_SUFFIX)
MYPACKAGE_SOURCE := $(SRCDIR)/$(MYPACKAGE).$(MYPACKAGE_SUFFIX)
MYPACKAGE_DIR := $(BUILDDIR)/$(MYPACKAGE)
MYPACKAGE_LICENSE := 0BSD
MYPACKAGE_LICENSE_FILES := file://LICENSE;md5=60b725f10c9c85c70d97880dfe8191b3
# License Report

```bash
$ ptxdist make license-report
```

## Contents

<table>
<thead>
<tr>
<th>Number</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>dosfstools</td>
<td>42</td>
</tr>
<tr>
<td>20.1</td>
<td>COPYING</td>
<td>42</td>
</tr>
<tr>
<td>20.2</td>
<td>src_fskfat.c</td>
<td>54</td>
</tr>
<tr>
<td>21</td>
<td>e2fsprogs</td>
<td>55</td>
</tr>
<tr>
<td>21.1</td>
<td>NOTICE</td>
<td>55</td>
</tr>
<tr>
<td>21.2</td>
<td>lib_uid_gen_uid.c</td>
<td>71</td>
</tr>
<tr>
<td>22</td>
<td>expat</td>
<td>72</td>
</tr>
<tr>
<td>22.1</td>
<td>COPYING[automatically found]</td>
<td>72</td>
</tr>
<tr>
<td>23</td>
<td>gcclibs</td>
<td>73</td>
</tr>
<tr>
<td>23.1</td>
<td>COPYING3</td>
<td>73</td>
</tr>
<tr>
<td>23.2</td>
<td>COPYING.RUNTIME</td>
<td>85</td>
</tr>
<tr>
<td>24</td>
<td>gdbserver</td>
<td>87</td>
</tr>
<tr>
<td>24.1</td>
<td>COPYING</td>
<td>87</td>
</tr>
<tr>
<td>24.2</td>
<td>COPYING3</td>
<td>93</td>
</tr>
<tr>
<td>24.3</td>
<td>COPYING.LIB</td>
<td>105</td>
</tr>
<tr>
<td>24.4</td>
<td>COPYING3</td>
<td>114</td>
</tr>
<tr>
<td>25</td>
<td>glib</td>
<td>118</td>
</tr>
<tr>
<td>25.1</td>
<td>COPYING[automatically found]</td>
<td>118</td>
</tr>
<tr>
<td>26</td>
<td>glibc</td>
<td>128</td>
</tr>
<tr>
<td>26.1</td>
<td>COPYING</td>
<td>128</td>
</tr>
<tr>
<td>26.2</td>
<td>COPYING.LIB</td>
<td>134</td>
</tr>
<tr>
<td>26.3</td>
<td>LICENSES</td>
<td>143</td>
</tr>
</tbody>
</table>

## 53 openssh

**Package:** openssh 8.3p1  
**License:** BSD AND BSD-2-Clause AND BSD-3-Clause AND MIT AND Beowave AND BSC  
**Flags:** attribution  
**URL:**  
https://ftp.halifax.ncl.ac.uk/openbsd/OpenSSH/portable/openssh-8.3p1.tar.gz  
https://mirror.lsu.edu/debian/pft/OpenBSD/OpenSSH/portable/openssh-8.3p1.tar.gz  
**MDS:** 68d7eb27b2672153ca47402f6489a1af  

![Dependency tree for openssh](openssh_zlib_openssl)

## 53.1 LICENCE

This file is part of the OpenSSH software.

The licences which components of this software fall under are as follows. First, we will summarize and say that all components are under a BSD licence, or a licence more free than that.

OpenSSH contains no GPL code.

1. Copyright (c) 1995 Tatu Torman <tjc@cs.hut.fi>, Espoo, Finland  
   * All rights reserved  
   * As far as I am concerned, the code I have written for this software  
   * can be used freely for any purpose. Any derived versions of this  
   * software must be clearly marked as such, and if the derived work is  
   * incompatible with the protocol described in the RFC files, it must be  
   * called by a name other than "ssh" or "Secure Shell".  

25/33
Reproducible Builds

- Ideally: same source input == same binary output
- Set SOURCE_DATE_EPOCH variable during build
  - Packages’s build systems need to support it
BSP Documentation

- reStructuredText + sphinx
- HTML and PDF output

```bash
~/projects/distrokit $ ls -l doc
guru.rst
hardware_mips_qemu.rst
hardware_rpi_raspi1.rst
hardware.rst
hardware_v7a_beaglebone_black.rst
hardware_v7a_beaglebone_white.rst
hardware_v7a_nitrogen6x.rst
hardware_v7a_qemu.rst
hardware_v7a_raspi2.rst
hardware_v7a_raspi3.rst
hardware_v7a_riot.rst
hardware_v7a_udoo_neo.rst
hardware_v8a_espressobin.rst
hardware_x86_64_qemu.rst
index.rst
intro.rst
```
Development Helpers

NFS Root

• Export the BSPs sysroot on the devel host via NFS
  - Set correct access rights etc.

• Use vim on the devel host instead of the target :-)

• No rebuilding, reflashing, and rebooting needed

```
~/projects/my-bsp $ ptxdist nfsroot

Mount rootfs with nfsroot=/root,v3,tcp,port=2049,mountport=2049
```
Development Helpers

GDB Wrapper

• Use with gdbserver or coredumps
• Debug on the host, not on the target

```bash
~/projects/my-bsp $ ptxdist gdb -quiet platform-v7a/root/usr/bin/mount
Reading symbols from platform-v7a/root/usr/bin/mount...
Reading symbols from platform-v7a/root/usr/lib/debug/.build-id/16/7c142341573f764667b5d22f6ba14aa9e78f15.debug...
(gdb) print main
$1 = {int (int, char **)} 0x2618 <main>
(gdb) list main
574    return 0;
575     return ret;
576   }
577
578 int main(int argc, char **argv)
579 {
580    int c, rc = MNT_EX_SUCCESS, all = 0, show_labels = 0;
581    struct libmnt_context *ctx;
582    struct libmnt_table *fstab = NULL;
583    char *srcbuf = NULL;
(gdb) 
```
Summary

• Menu interface
• Stable and known technology under the hood
• Focus on reproducibility
  – Pin down variable space as much as possible
  – Solve conflicts early
• Code Signing Infrastructure
• Development helpers (NFS root, GDB)
Trying It Out

**DistroKit**

- preconfigured BSP for a variety of dev boards (and qemu)
- [https://git.pengutronix.de/cgit/DistroKit](https://git.pengutronix.de/cgit/DistroKit)
References

Icons

• FontAwesome (https://fontawesome.com), CC BY 4.0

Tux Logo

• Attribution: Larry Ewing <lewing@isc.tamu.edu> and The GIMP
Thanks for listening!

Questions?