Void Linux

We heard you like little endian
Firstly, who am I?

- A programmer from the Czech Republic
- Involved in open source since 2007
- Background in graphics toolkits (EFL), programming languages, game development
- Currently the primary maintainer for POWER architecture support in Void (and other several things)
What is Void Linux?

- Independent Linux distro with its own package manager
- Focus on simplicity and pragmatism
- Rolling release, stable software, binary based
- Portable (also on x86, ARM, MIPS), choice of glibc and musl, runit, “ports” (xbps-src), cross-compilation
- Low barrier of entry, open and inclusive, informal
Void on POWER vs other distros

- Desktop/workstation focus, not server
- Both little and big endian support
- ELFv2 ABI on big endian glibc
- 4K pages kernel by default
- Wider desktop software support
Quick history of Void on POWER

- Late 2018 – initial port on Talos 2 Lite (ppc64le)
- Early 2019 – gradual expansion of package coverage
- 64-bit BE musl port, 32-bit ports
- April 2019 – 64-bit BE glibc port (ELFv2 ABI)
- Late 2019 – complete repository coverage
- OpenPOWER Summit EU talk
- November 2019 – commit bit in upstream
News since then
Java bootstrap

- OpenJDK 8 and 11 in repositories
- Needs itself to build (one version older)
- Re-introduction of GCC6 (last version with GCJ)
- Bootstrap packages for JDK 7, 9, 10
- JITed on ppc64(le), slow on ppc32 (zero backend)
Go bootstrap

- Only on ppc64le
- Using binary bootstrap (official binaries)
- gccgo unreliable (gcompat on musl)
- BE support – wrong ABI and needs POWER8
- 32-bit support – missing entirely
Haskell bootstrap

- Self-built binary bootstrap snapshots
- Includes musl (figured out cross-builds)
- BE missing for now (ELFv2 patch made though)
- Big endian bugs prevent functional bindist
- More target support later for 8.10
Other languages

- Greater Common Lisp support
- SBCL finally in repos (for now LE only)
- Bootstrapped with ECL
- Preparing a Clisp update
- D, Zig and some others still missing
LibreSSL performance

- Void uses LibreSSL instead of OpenSSL
- Assembly only on x86 and some 32-bit ARM
- Inferior performance and no hardware crypto support
- Solution: import assembly bits from OpenSSL upstream
LibreSSL performance

- [https://github.com/q66/libressl-portable-asm](https://github.com/q66/libressl-portable-asm)
- Now shipping in Void
- Also adds Aarch64 assembly support and improves ARM
- Up to 20x performance increase
- Especially in hw crypto (ghash/AES etc.) on POWER9
Chromium support

• No official upstream support
• Downstream patches available
• Shipping in Void since 84
• Currently only browser engine with JIT support
• Works on musl as well
Electron support

- We have systemwide Electron now
- For now enables VS Code, Element, Rocket.Chat-Desktop
- Only LE, musl works
- Ugly build system workarounds
AMD GPU support

- Problem: AMDGPU DC
- Haphazard hardware floating point usage
- Page size dependency
- Upstream support only in recent kernels
- Need backports – provided down to 5.4
Cross-toolchain rework

- Code duplication for cross-toolchain build templates
- Solution: introduce a common build-style
- Each cross-toolchain template is now ~50 lines
- Extra goodies: glibc cross on musl hosts, etc.
- Unified configure args and so on for all
- No more dirty masterdirs
Infrastructure status
Infrastructure status

- Still a Talos 2 Lite (18C, 128G RAM) in my bedroom
- Big endian builds in a VM (KVM-HV)
- Little endian builds on bare metal
- Primary mirror – separate server (in Chicago, IL)
- 10G network, 2TB storage
Infrastructure status

- Several other mirrors provided by community
- Could use faster build hardware
- Could use hardware not in my bedroom
- Primary mirror should migrate to build machine
- More build automation necessary – WIP
32-bit little endian
32-bit little endian

- Why not?
- Officially does not exist
- In practice, this just works though
- Pretty basic bringup process
- Made easy by xbps-src
32-bit little endian

- No native support for 32-bit LE in Linux
- Works as a chroot on 64-bit kernel though
- But not out of box: fixes are needed
- Fix 32-bit compat in kernel first
- Several issues
32-bit LE kernel compat

- Entering signal handlers clears MSR_LE
- vDSO32 disabled by default
- Certain vDSOs broken on LE – disabled those
- Syscalls with 64-bit args: wrong byte order
- All very trivial patches
- Things “run” without them
32-bit LE kernel compat

- But they crash at random points
- After patching, no more known issues
- Fixes are being upstreamed
- VDSO32 will be fixed by moving it to C
- Backports in all Void kernels (4.19+)
Bootstrapping 32-bit LE

- First: create build and cross xbps-src profiles
- Second: create cross-toolchains for 32-bit LE targets
- Minor patching in glibc and musl needed
- Adjustments around xbps-src needed
Bootstrapping 32-bit LE

- Try cross-compiling base-chroot
- Minimal set of packages for the build container
- Fix any errors along the way
- Relatively few and mostly trivial
- Binary-bootstrap a build container after that
- Then rebuild every package (no cross)
Bootstrapping 32-bit LE

- Now we can build the other base metapackages
- Afterwards, other software can be built
- Usual “fix whatever pops up” approach applies
- Most things generally just work
- Could probably function on 32-bit userland
32-bit LE: the Buts

- No official support in glibc
- Dynlinker, symvers etc. shared with 32-bit BE
- Going to propose an official port later
- Same SVR4 ABI as 32-bit BE
- Musl pretty much just works
32-bit LE: the Why

- Because we could, of course
- Easier emulation of 32-bit x86
- Port LE Linux to G3s/G4s? :)
- Or use BE Linux and swizzle syscall data?
- For now, strictly POWER8+ though
32-bit LE: the obstacles

- LLVM support
- Recently upstreamed by the FreeBSD folks
- Backported to Void
- Rust support
- Blocks a fair amount of userland right now
- WIP patches available
32-bit LE: the obstacles

• Proper glibc port needs to be made
• Use the opportunity to improve the ABI?
• Use 64-bit long doubles later (ditch ibm128)
• Choose a unique dynlinker name and fresh symvers
• Fix up our 32-bit BE ports too?
32-bit LE: the results

- Testing repos now available (glibc and musl)
- Can fully compile itself
- A portion of the repo packaged
- Core userland, dev tools, LLVM, Mesa, SDL...
- Initial port of box86
Box86

- Linux x86 emulator developed for ARM devices
- Needs 32-bit little endian host
- Initial PowerPC port now
- Can run glxgears
- Can run Unreal Tournament
- Fairly slow though… no dynarec
The future
Infrastructure

- Official repo at some point?
- 64LE should be easy enough
- 64BE and 32-bit have some issues with server software
- They also need clearing up the ABI situation
ELFv2 BE ABI

• We will need official support in glibc
• Right now things work “by accident”
• Need unique dynlinker name and symvers
• Upstream said they wouldn’t be against
• But, need to formalize the ABI
ELFv2 BE ABI

- IBM thinks VSX requirement is a part of the ABI
- Solution: Formalize “ELFv2 Legacy-Compatible” ABI
- To be shared by the new glibc port, musl and FreeBSD
- ELFv2 minus POWER8/VSX requirement
ELFv2 BE ABI

• Extension of the official spec
• No VSX nor VMX requirement
• 64-bit long doubles – ditch ibm128, and do not use ieee128 (needs vector registers for passing)
• Coming during 2021
Other things

- Upstream as many patches as possible
- Write a new installer
- Work on enablement of new software
- Work with upstreams on adding support
Conclusion

- Lots of work has been done in a year and half
- There is still plenty more to be done
- Join us on IRC: #voidlinux-ppc @ freenode
- For detailed statistics, visit https://repo.voidlinux-ppc.org/stats.html
Questions time
And thanks for listening!

Daniel Kolesa
Twitter: @octaforge
daniel@octaforge.org
https://voidlinux-ppc.org
FOSDEM 2021