a whirlwind tour of portability

kyle mcmartin
kyle@redhat.com

about me

- debian
- fedora
- rhel
- linux, toolchain, glibc, etc.

linux portability

initially 386 only, non-portable

then came the alpha
64-bit, little endian

then sparc
big-endian, more complicated caching

and then mips, arm, and now
the most portable kernel

signedness

char signedness ends up being a hilarious bug

```
char c = -1;
if (c < 0)
```

signedness of char is not defined

ppc, arm, s390 : unsigned
basically everything else: signed

compiler warning, but who looks at those?

-fsigned-char to CFLAGS as a temporary fix
work with upstream

endianness

aka byte-order

filesystems & block devices

superblock magic

want to be able to use the same filesystem on multiple machines

pick a byte-order and stick to it

byte-swap if need be

networking

tcp/ip is big-endian

pci is little-endian

native usually native

alignment

addr % width = 0
natural alignment

3 ways of handling this

fix up in hardware, natively slow

trap, fix in software
slower

return bad data
(arm < v6)

but they fixed that

implications of alignment

alignment of 64-bit
x86, x86_64

has syscall ABI implications

unless you guarantee proper alignment,
u64 will not be aligned correctly given the same struct

calling conventions

register usage

64-bit even-odd register pairs

reduces number of args available for syscall

have to write syscall wrapper to handle cases in glibc and kernel

page size issues

4K pages standard, 8K on some 64-bit platforms

64K pages on new platforms

64K doesn't fit in a u16

lots of hardware has 16-bit size registers

64K PAGE_SIZE, means truncated write

other hilarious problems

kernel is built “freestanding”

but gcc relies on callouts for some operations

especially integer division (signed/unsigned)
64-bit

common problem with 64-bit variables
i686 had it with PAE
arm getting it with lpae now

division operations on dma_addr_t

code generation issues

lots of tools need to JIT code these days

in the bad olde days
everyone wrote custom code gen

now everyone bundles llvm

good: standard
bad: usually only single llvm versions supported
may not be what fedora ships

portability issues, code size on risc

limited branch distance
pcrel usually limited to +/- 1MB

standardized jit toolchain makes this simpler
don't need to fix multiple places