Tübix 2019

The End of Time
19 years to go

Arnd Bergmann, Linaro

Overview

- Background: The problem we are solving
- Changes merged so far
- Ongoing changes

Background

Background: time_t in Unix

- typedef long time_t;
- time_t start=0:
  1970-01-01, 00:00:00 UTC
- 32-bit TIME_T_MAX:
  2038-01-19, 03:14:07 UTC

The fix: 64-bit time_t

- typedef long long time_t;
- 64-bit TIME_T_MAX: 292,277,026,597-12-04 14:15:28

Why we care: 32-bit kernels

- Widespread use:

```
$ find arch/arm/boot/dts/  -name \*.dts | wc -l
   1099
$ find arch/arm64/boot/dts/  -name \*.dts | wc
-l
    193
```

- Sometimes long service lives

32 bit devices with 20+ year life

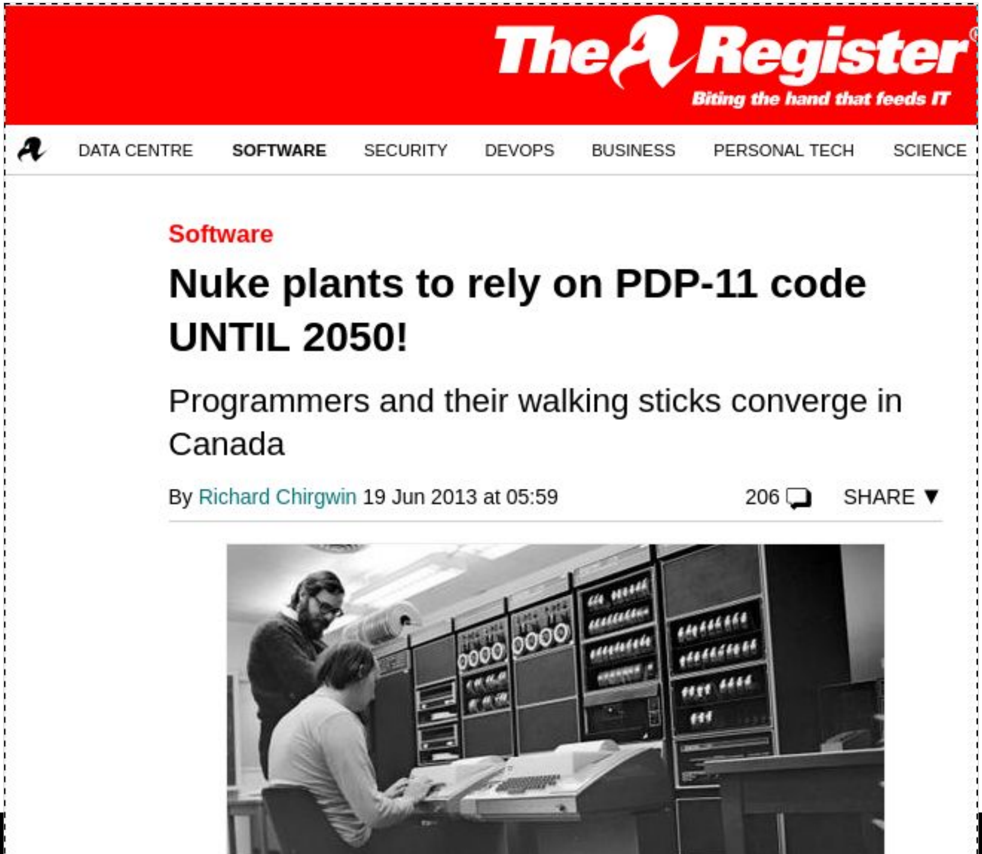
tübix

32 bit devices with 20+ year life

tübix

32 bit devices with 20+ year life

Some software lives even longer


The Register
Biting the hand that feeds IT

DATA CENTRE SOFTWARE SECURITY DEVOPS BUSINESS PERSONAL TECH SCIENCE

Software

Nuke plants to rely on PDP-11 code UNTIL 2050!

Programmers and their walking sticks converge in Canada

By Richard Chirgwin 19 Jun 2013 at 05:59 206 SHARE ▼

Ongoing work in the kernel

Industrial product lifecycle

- Development starts on proven hardware with long service life (e.g. NXP i.MX6)
- Several years until first deployment
- 5-10 years active marketing
- Customer buys 15 year old embedded system
- Expect 10+ years of active use

32 bit user space

- nonportable legacy applications
- memory size limitations
- common software
  - ARMv6 Raspbian

32 bit interfaces

- Network protocols
  - Shared key expiration
- File systems
  - On-disk inode timestamps
- File formats
  - utmp
  - cpio

32 bit interfaces

- Hardware/Firmware
  - Real-time clock
  - SCSI adapters
  - PTP network adapters

```
SHMEM2_WR(bp, drv_info.epoc,
          (u32)ktime_get_real_seconds());
```

Ongoing work in the kernel

```
commit 361a3bf00582469877f8d18ff20f1efa6b781274
Author: John Stultz <john.stultz@linaro.org>
Date:   Wed Jul 16 21:03:58 2014 +0000

    time64: Add time64.h header and define struct timespec64

    Define the timespec64 structure and standard helper functions.

    [ tglx: Make it 32bit only. 64bit really can map timespec to timespec64 ]

    Signed-off-by: John Stultz <john.stultz@linaro.org>
    Signed-off-by: Thomas Gleixner <tglx@linutronix.de>
    Signed-off-by: John Stultz <john.stultz@linaro.org>

diff --git a/include/linux/time64.h b/include/linux/time64.h
new file mode 100644
index 000000000000..e7b499e1cd79
--- /dev/null
+++ b/include/linux/time64.h
@@ -0,0 +1,162 @@
+#ifndef _LINUX_TIME64_H
+#define _LINUX_TIME64_H
+
+#include <uapi/linux/time.h>
+
+typedef __s64 time64_t;
+
+/*
+ * This wants to go into uapi/linux/time.h once we agreed about the
```

Eliminating all 32-bit time_t uses

- Many hundreds of drivers patched since 2014
- Core timekeeping code, 2014-2015
- Core file system code, 2012-2018
- System calls, ongoing

Eliminating all 32-bit time_t uses

- Change time* to ktime_t
  - Also helps with accuracy
- Change time* to jiffies
  - Also makes code faster
- Change time_t to time64_t
- Change timespec/timeval to timespec64
- Change CLOCK_REALTIME to CLOCK_MONOTONIC
  - Also helps with leap seconds, NTP

Fixing ioctl interfaces

```
#define _IOC(dir,type,nr,size) \
        (((dir)  << _IOC_DIRSHIFT) | \
         ((type) << _IOC_TYPESHIFT) | \
         ((nr)   << _IOC_NRSHIFT) | \
         ((size) << _IOC_SIZESHIFT))

#define _IOR(type,nr,size)      _IOC(_IOC_READ,(type),(nr),(sizeof(size)))
#define _IOW(type,nr,size)      _IOC(_IOC_WRITE,(type),(nr),(sizeof(size)))
#define _IOWR(type,nr,size)
_IOC(_IOC_READ|_IOC_WRITE,(type),(nr),(sizeof(size)))

#define PPGETTIME    _IOR(PP_IOCTL, 0x95, struct timeval)

#define PPPIOCGIDLE  _IOR('t', 63, struct ppp_idle) /* get idle time */
```

Fixing ioctl interfaces

```
@@ -743,10 +744,17 @@ static long ppp_ioctl(struct file *file, unsigned int
cmd, unsigned long arg)
                err = 0;
                break;

-       case PPPIOCGIDLE:
-               idle.xmit_idle = (jiffies - ppp->last_xmit) / HZ;
-               idle.recv_idle = (jiffies - ppp->last_recv) / HZ;
-               if (copy_to_user(argp, &idle, sizeof(idle)))
+       case PPPIOCGIDLE32:
+                idle32.xmit_idle = (jiffies - ppp->last_xmit) / HZ;
+                idle32.recv_idle = (jiffies - ppp->last_recv) / HZ;
+                if (copy_to_user(argp, &idle32, sizeof(idle32)))
+               err = 0;
+               break;
+
+       case PPPIOCGIDLE64:
+               idle64.xmit_idle = (jiffies - ppp->last_xmit) / HZ;
+               idle64.recv_idle = (jiffies - ppp->last_recv) / HZ;
+               if (copy_to_user(argp, &idle32, sizeof(idle32)))
```

Fixing ioctl interfaces

```
#define _IOC(dir,type,nr,size) \
        (((dir)  << _IOC_DIRSHIFT) | \
         ((type) << _IOC_TYPESHIFT) | \
         ((nr)   << _IOC_NRSHIFT) | \
         ((size) << _IOC_SIZESHIFT))

#define _IOR(type,nr,size)      _IOC(_IOC_READ,(type),(nr),(sizeof(size)))
#define _IOW(type,nr,size)      _IOC(_IOC_WRITE,(type),(nr),(sizeof(size)))
#define _IOWR(type,nr,size)
_IOC(_IOC_READ|_IOC_WRITE,(type),(nr),(sizeof(size)))

#define PPGETTIME    _IOR(PP_IOCTL, 0x95, struct timeval)

#define PPPIOCGIDLE  _IOR('t', 63, struct ppp_idle) /* get idle time */

#define SIOCGSTAMP   0x8906
```

Fixing ioctl interfaces

```
-#define SIOCGSTAMP        0x8906
+#define SIOCGSTAMP_OLD    0x8906
+/*
+ * the timeval/timespec data structure layout is defined by libc,
+ * so we need to cover both possible versions on 32-bit.
+ */
+/* Get stamp (timeval) */
+#define SIOCGSTAMP_NEW   _IOR(SOCK_IOC_TYPE, 0x06, long long[2])
+
+#if __BITS_PER_LONG == 64 || (defined(__x86_64__) && defined(__ILP32__))
+/* on 64-bit and x32, avoid the ?: operator */
+#define SIOCGSTAMP     SIOCGSTAMP_OLD
+#else
+#define SIOCGSTAMP     ((sizeof(struct timeval))  == 8 ? \
+                        SIOCGSTAMP_OLD   : SIOCGSTAMP_NEW)
+#endif
```

Other driver interfaces: read()

```
int input_event_to_user(char __user *buffer,
                        const struct input_event *event)
{
        if (in_compat_syscall() && !COMPAT_USE_64BIT_TIME) {
                struct input_event_compat compat_event;

                compat_event.sec = event->input_event_sec;
                compat_event.usec = event->input_event_usec;

                if (copy_to_user(buffer, &compat_event,
                                 sizeof(struct input_event_compat)))
                        return -EFAULT;
        } else {
                if (copy_to_user(buffer, event, sizeof(struct input_event)))
                        return -EFAULT;
        }

        return 0;
}
```

Other driver interfaces: read()

```
struct input_event {
#if (__BITS_PER_LONG != 32 || !defined(__USE_TIME_BITS64)) &&
!defined(__KERNEL)
        struct timeval time;
#define input_event_sec time.tv_sec
#define input_event_usec time.tv_usec
#else
        __kernel_ulong_t __sec;
        __kernel_ulong_t __usec;
#define input_event_sec  __sec
#define input_event_usec __usec
#endif
        __u16 type;
        __u16 code;
        __s32 value;
};
```

Other driver interfaces: mmap()

```
struct snd_pcm_mmap_status {
        snd_pcm_state_t state;          /* RO: state - SNDRV_PCM_STATE_XXXX
*/
        int pad1;                       /* Needed for 64 bit alignment */
        snd_pcm_uframes_t hw_ptr;       /* RO: hw ptr (0...boundary-1) */
-       struct timespec tstamp;         /* Timestamp */
+       struct snd_monotonic_timestamp tstamp; /* Timestamp */
        snd_pcm_state_t suspended_state; /* RO: suspended stream state */
-       struct timespec audio_tstamp;   /* from sample counter or wall clock
*/
+       struct snd_monotonic_timestamp audio_tstamp;    /* from sample
counter
+                                                          or wall clock */
 };

 struct snd_pcm_mmap_control {
```

Virtual File System layer

- First posted by Arnd Bergmann in 2014
- Second try: Deepa Dinamani, 2016
- Completed by Deepa in 2018
- statx() syscall by David Howells
- utimes() syscall: WIP
- File systems mostly converted
  - Missing: NFS, XFS, HFS, AFS
- Some file systems still not y2038 safe
  - XFS, ext3, coda

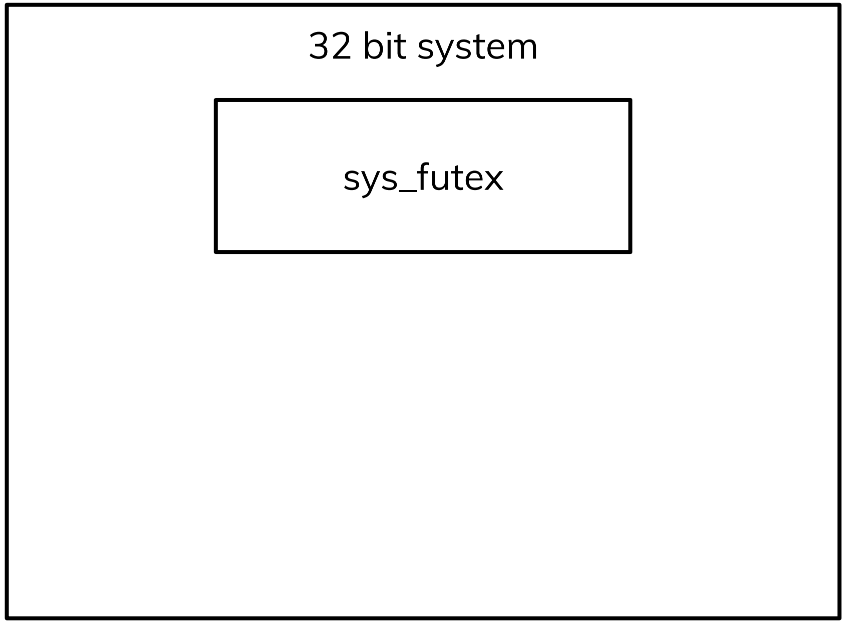
System calls

- Internal implementation done
- New entry points for 32-bit
  50% done in 4.18, 80% in 4.20
- Need to discuss some APIs:
  - clock_adjtimex
  - getrusage, wait4
  - getitimer/setitimer

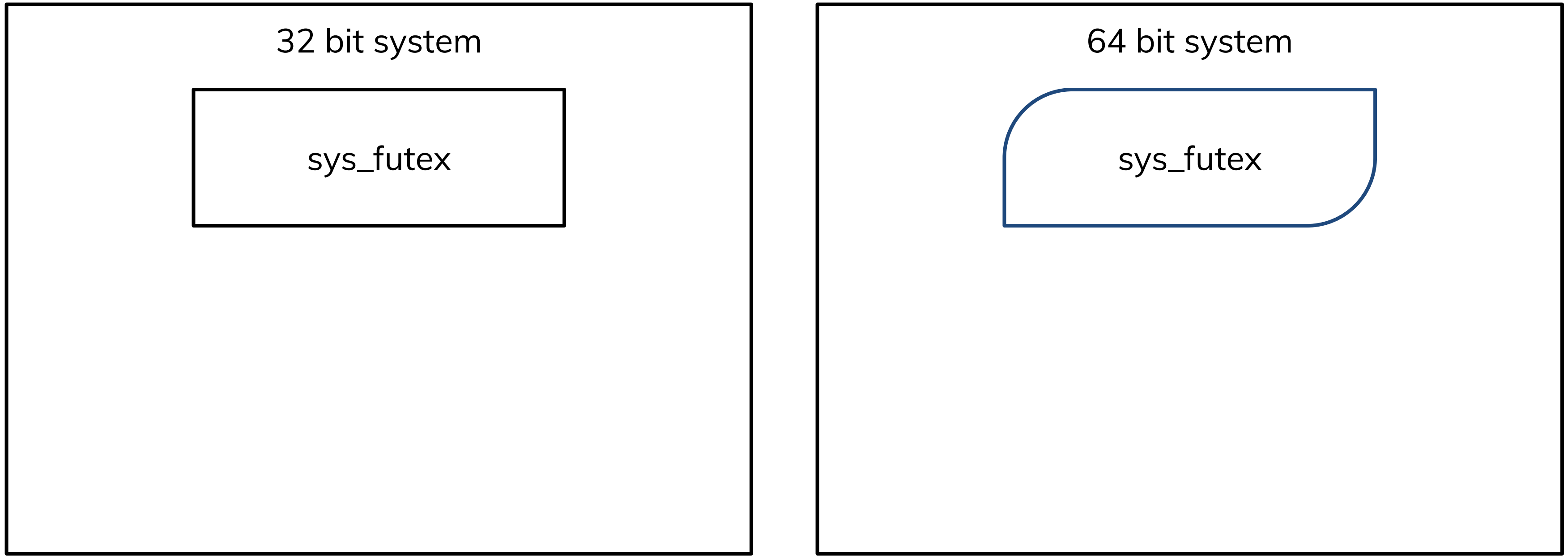
System calls: method

- Change normal syscall to 64-bit time_t interface
- Reuse 32-bit compat syscalls
- rename compat_time_t to old_time32_t
- rename compat_sys_foo() to sys_foo_time32()

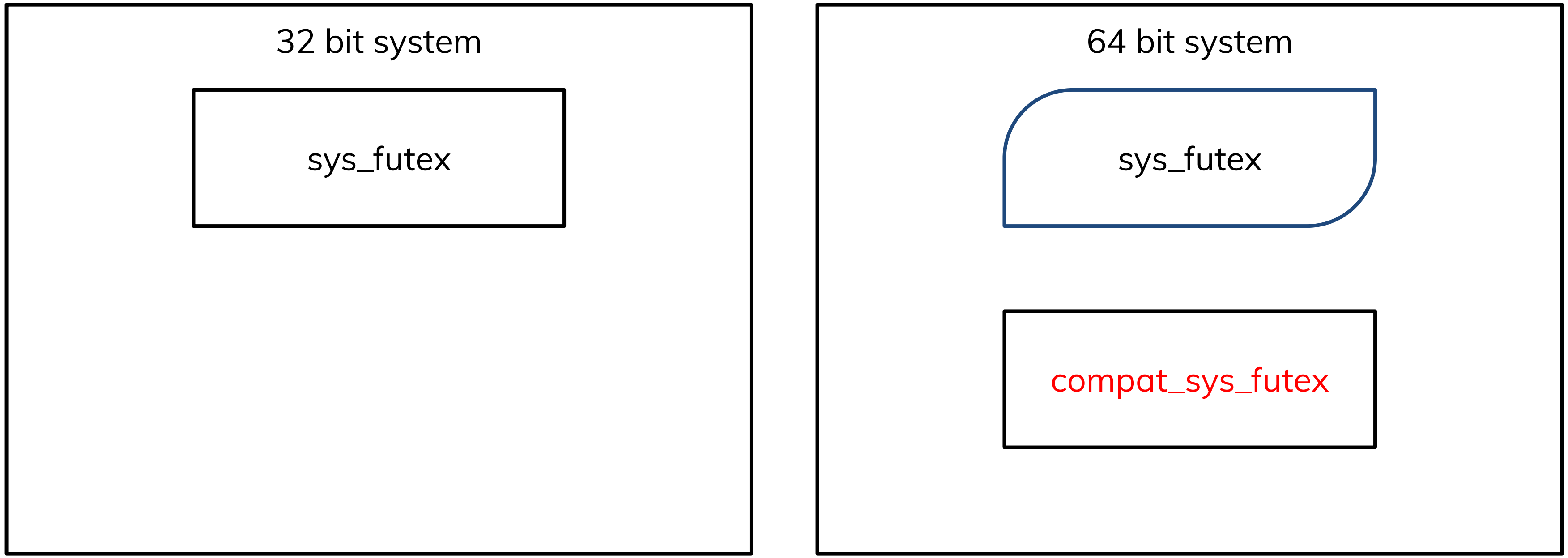
System calls: method



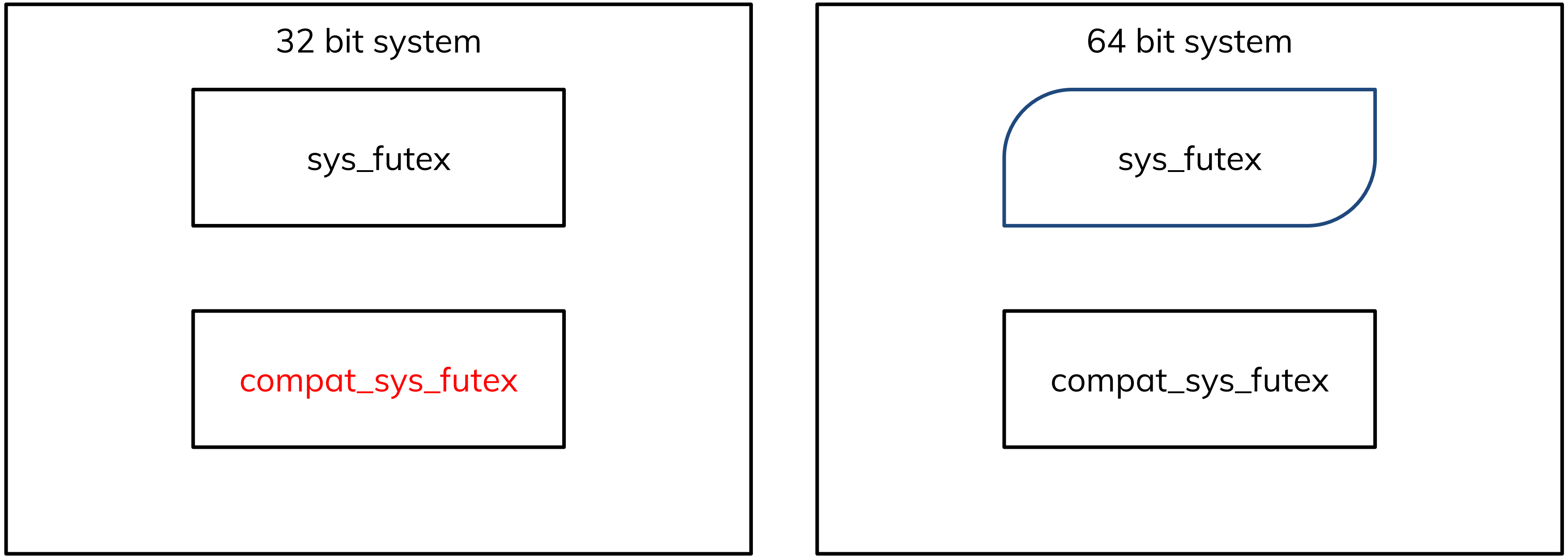
System calls: method



System calls: method



System calls: method

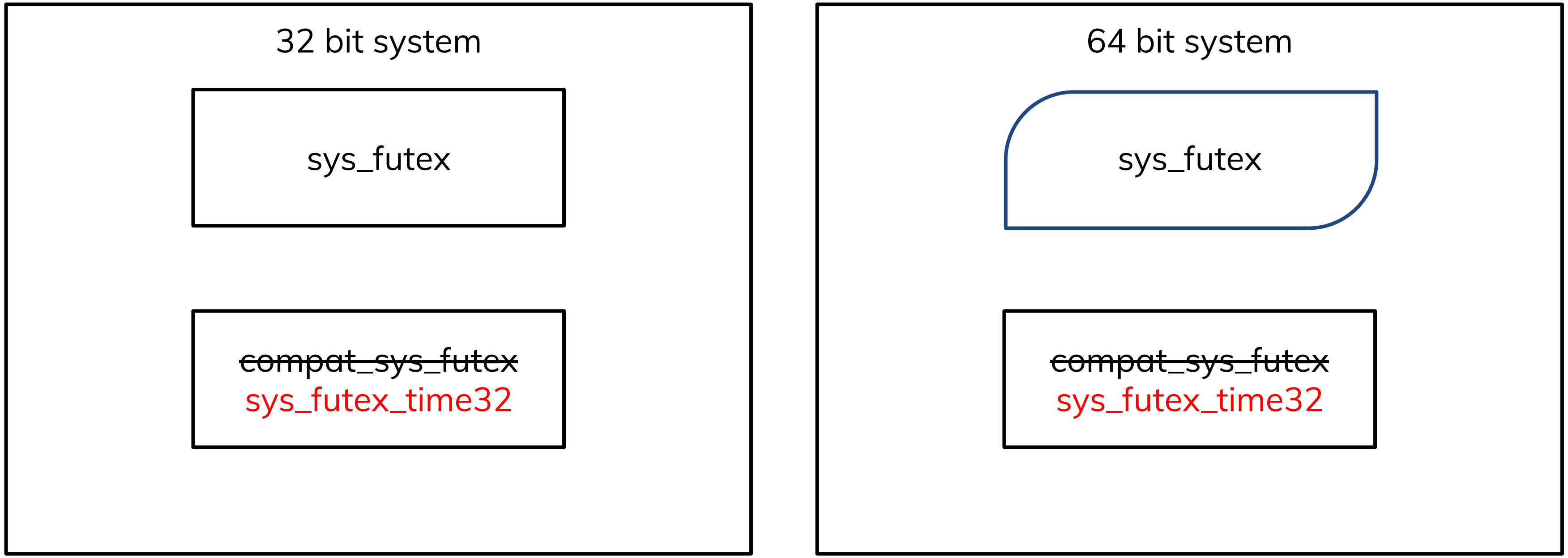


System calls: method

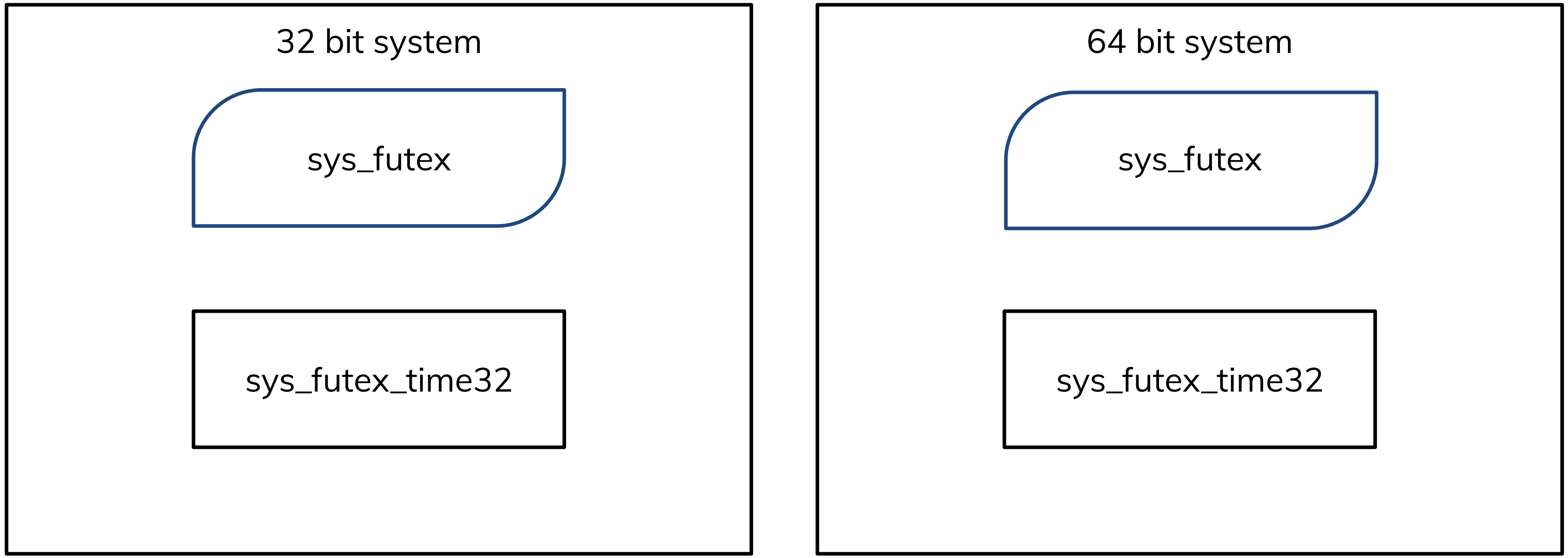
```
@@ -173,10 +173,10 @@ COMPAT_SYSCALL_DEFINE3(get_robust_list, int, pid,
        return ret;
 }

-#ifdef CONFIG_COMPAT
-COMPAT_SYSCALL_DEFINE6(futex, u32 __user *, uaddr, int, op, u32, val,
-                struct compat_timespec __user *, utime, u32 __user *, uaddr2,
+#ifdef CONFIG_COMPAT_32_BIT_TIME
+SYSCALL_DEFINE6(futex_time32, u32 __user *, uaddr, int, op, u32, val,
+                struct old_timespec32 __user *, utime, u32 __user *, uaddr2,
                u32, val3)
 {
        struct timespec ts;
```

System calls: method



System calls: method



System calls: method

```
--- a/kernel/futex.c
+++ b/kernel/futex.c
@@ -3558,10 +3558,10 @@ long do_futex(u32 __user *uaddr, int op, u32 val,
ktime_t *timeout,


 SYSCALL_DEFINE6(futex, u32 __user *, uaddr, int, op, u32, val,
-               struct timespec __user *, utime, u32 __user *, uaddr2,
+               struct __kernel_timespec __user *, utime, u32 __user *,
uaddr2,
                u32, val3)
 {
-       struct timespec ts;
+       struct timespec64 ts;
        ktime_t t, *tp = NULL;
        u32 val2 = 0;
        int cmd = op & FUTEX_CMD_MASK;
```

System calls: method

include/linux/time64.h

```
typedef __s64 time64_t;

#ifndef CONFIG_64BIT_TIME
#define __kernel_timespec timespec
#endif


#include <uapi/linux/time.h>
```

include/uapi/linux/time.h

```
typedef long long __kernel_time64_t;

#ifndef __kernel_timespec
struct __kernel_timespec {
        __kernel_time64_t       tv_sec;
        long long               tv_nsec;
};
#endif
```

Affected syscalls, deprecated

- time
- stime
- gettimeofday
- settimeofday
- adjtimex
- nanosleep
- alarm
- select
- old_select
- io_getevents

- utime
- utimes
- futimensat
- oldstat
- oldlstat
- oldfstat
- newstat
- newlstat
- newfstat
- newfstatat

- stat64
- lstat64
- fstat64
- fstatat64
- wait4

Affected syscalls, deprecated

- ~~time~~
- ~~stime~~
- ~~gettimeofday~~
- ~~settimeofday~~
- adjtimex
- ~~nanosleep~~
- ~~alarm~~
- select
- old_select
- io_getevents

- ~~utime~~
- ~~utimes~~
- ~~futimensat~~
- ~~oldstat~~
- ~~oldlstat~~
- ~~oldfstat~~
- ~~newstat~~
- ~~newlstat~~
- ~~newfstat~~
- ~~newfstatat~~

- ~~stat64~~
- ~~lstat64~~
- ~~fstat64~~
- ~~fstatat64~~
- wait4

Affected syscalls, deprecated

- ~~time~~
- ~~stime~~
- ~~gettimeofday~~
- ~~settimeofday~~
- adjtimex
- ~~nanosleep~~
- ~~alarm~~
- ~~select~~
- ~~old_select~~
- ~~io_getevents~~
- ~~utime~~
- ~~utimes~~
- ~~futimensat~~
- ~~oldstat~~
- ~~oldlstat~~
- ~~oldfstat~~
- ~~newstat~~
- ~~newlstat~~
- ~~newfstat~~
- ~~newfstatat~~
- ~~stat64~~
- ~~lstat64~~
- ~~fstat64~~
- ~~fstatat64~~
- wait4

Affected syscalls, need replacement

- clock_gettime
- clock_settime
- clock_adjtime
- clock_getres
- clock_nanosleep
- getitimer
- setitimer
- timer_gettime
- timer_settime
- timerfd_gettime
- timerfd_settime
- pselect6
- ppoll
- io_pgetevents
- recvmmsg
- mq_timedsend
- mq_timedreceive
- semtimedop
- msgctl
- semctl
- shmctl
- utimensat
- rt_sigtimedwait
- futex
- sched_rr_get_interval
- getrusage
- wait4
- waitid
- sysinfo

Affected syscalls, need replacement

- ~~clock_gettime~~
- ~~clock_settime~~
- clock_adjtime
- ~~clock_getres~~
- ~~clock_nanosleep~~
- getitimer
- setitimer
- ~~timer_gettime~~
- ~~timer_settime~~
- ~~timerfd_gettime~~
- ~~timerfd_settime~~
- pselect6
- ppoll
- io_pgetevents
- recvmmsg
- ~~mq_timedsend~~
- ~~mq_timedreceive~~
- ~~semtimedop~~
- ~~msgctl~~
- ~~semctl~~
- ~~shmctl~~
- utimensat
- rt_sigtimedwait
- futex
- sched_rr_get_interval
- getrusage
- wait4
- waitid
- sysinfo

Affected syscalls, need replacement

- ~~clock_gettime~~
- ~~clock_settime~~
- clock_adjtime
- ~~clock_getres~~
- ~~clock_nanosleep~~
- getitimer
- setitimer
- ~~timer_gettime~~
- ~~timer_settime~~
- ~~timerfd_gettime~~
- ~~timerfd_settime~~
- ~~pselect6~~
- ~~ppoll~~
- ~~io_pgetevents~~
- ~~recvmmsg~~
- ~~mq_timedsend~~
- ~~mq_timedreceive~~
- ~~semtimedop~~
- ~~msgctl~~
- ~~semctl~~
- ~~shmctl~~
- ~~utimensat~~
- ~~rt_sigtimedwait~~
- ~~futex~~
- ~~sched_rr_get_interval~~
- getrusage
- wait4
- waitid
- sysinfo

Side note: time structures

```
struct __timespec64 {
  __time64_t  tv_sec;
#if __BYTE_ORDER == __BIG_ENDIAN
  int : 32;
#endif
  long        tv_nsec;
#if __BYTE_ORDER == __BIG_ENDIAN
  int padding;
#endif
};
```

Side note: time structures

```
struct __timespec64 {
  __time64_t  tv_sec;
#if __BYTE_ORDER == __BIG_ENDIAN
  int : 32;
#endif
  long        tv_nsec;
#if __BYTE_ORDER == __BIG_ENDIAN
  int padding;
#endif
};
```

Side note: time structures

```
struct __timespec64 {
  __time64_t  tv_sec;
#if __BYTE_ORDER == __BIG_ENDIAN
  int : 32;
#endif
  long        tv_nsec;
#if __BYTE_ORDER == __BIG_ENDIAN
  int padding;
#endif
};
```

Side note: time structures

```
struct __timespec64 {
  __time64_t  tv_sec;
#if __BYTE_ORDER == __BIG_ENDIAN
  int : 32;
#endif
  long        tv_nsec;
#if __BYTE_ORDER == __BIG_ENDIAN
  int padding;
#endif
};
```

Side note: time structures

```
struct __kernel_old_timeval {
          long tv_sec;
#ifdef __sparc_v9__
          int tv_usec;
          int padding;
#else
          long tv_usec;
#endif
};
```

Side note: time structures

```
struct __kernel_old_timeval {
          long tv_sec;
#ifdef __sparc_v9__
          int tv_usec;
          int padding;
#else
          long tv_usec;
#endif
};
```

Upcoming challenge:
User space

C library porting

GLIBC

- Design by Albert Aribaud
  - https://sourceware.org/glibc/wiki/Y2038ProofnessDesign
- Old kernel support
- Symbol versioning
- `gcc -D__USE_TIME_BITS64`

MUSL

- Experimental port by Arnd Bergmann
- Binary incompatible
- No compile time switch
- musl-2.x ?

Distro work needed

- 32-bit Embedded distros
  - OpenEmbedded,PTXdist OpenWRT, Buildroot,
  - Rebuild from source
- 32-bit Android
  - New incompatible ABI
- 32-bit Desktop
  - Debian, Fedora, …
  - Migration plan

- 64-bit distros
  - SLES, RHEL, Ubuntu, ...
  - Bug fixes only

32-bit distro needing rebuild

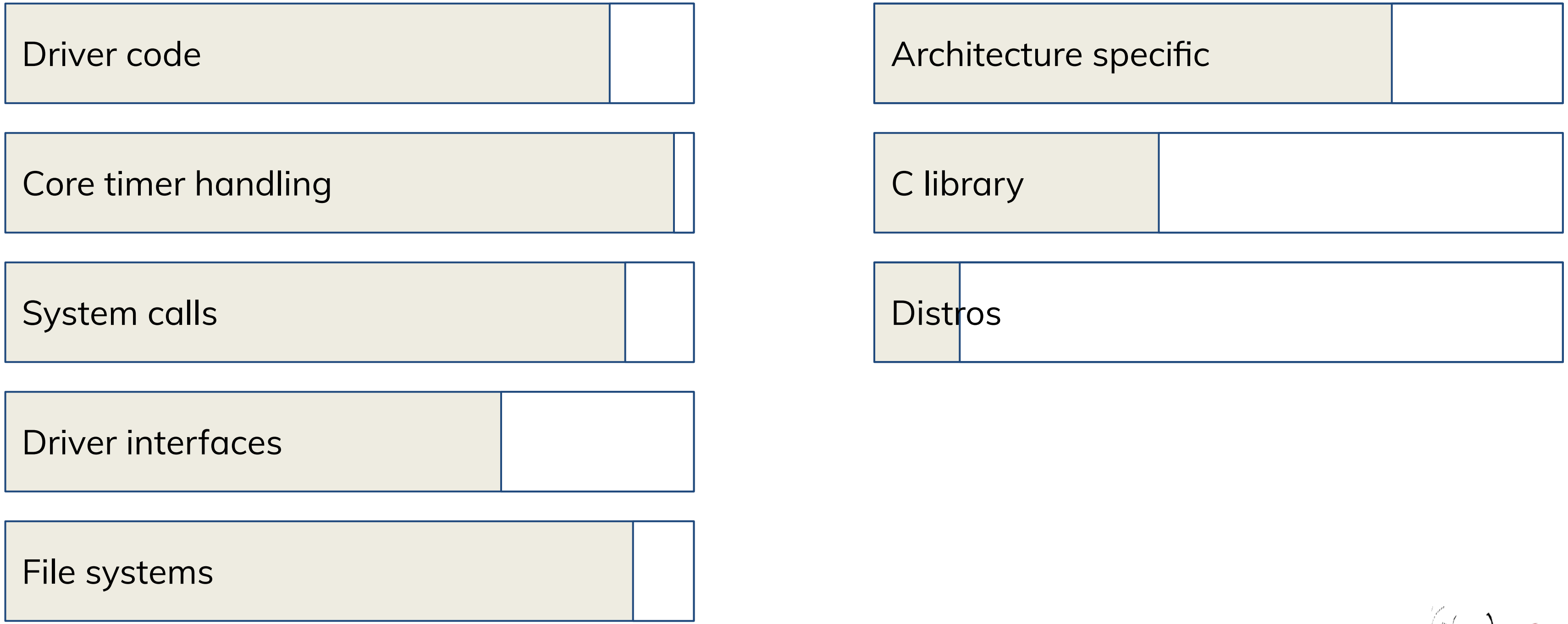
- Fedora
  - Primary: armhf
  - Secondary: x86-32, mips-el
- Debian
  - Official: armhf, armel, i386, mipsel
  - Other: sh4, m68k, or1k, powerpcspe

- Arch Linux
  - Inofficial: arm, x86-32
- Ubuntu Core (ARM):
  - Artik 5/10, Orange Pi 0, R-Pi 2
- openSUSE Leap:
  - armv6hl, armv7hl
- Raspbian

32-bit distro needing rebuild

- Fedora
  - Primary: armhf
  - Secondary: x86-32, mips-el
- Debian
  - Official: armhf, armel, i386, mipsel
  - Other: sh4, m68k, or1k, powerpcspe
- Arch Linux
  - Inofficial: arm, x86-32
- Ubuntu Core (ARM):
  - Artik 5/10, Orange Pi 0, R-Pi 2
- openSUSE Leap:
  - armv6hl, armv7hl
- Raspbian

Progress bar, v5.2



Questions?