LINUX-USER EMULATOR IN QEMU

Riku Voipio

WAYS TO USE QEMU

1. qemu-system-x86_64 -M accel=kvm ..
2. qemu-system-arm -M vexpress-a9 ...
3. **qemu-arm** ./busybox echo hi

HISTORY

```
Date: Sun, 23 Mar 2003 21:46:47 +0100
From: Fabrice Bellard <fabrice.bellard@free.fr>
Subject: [announce] QEMU x86 emulator version 0.1

Hi,

The first release of the QEMU x86 emulator is available at
http://bellard.org/qemu/. QEMU achieves a fast user space Linux x86
emulation on x86 and PowerPC Linux hosts by using dynamic translation.
Its main goal is to be able to run the Wine project on non-x86
architectures.

Fabrice.
```

HISTORY

- 2005: Qemu 0.7, mostly useful linux-user support
- - Nokia and others use linux-user to augment cross-compiling
- 2009: Qemu 0.10 gains KVM - focus changed away from linux-user
- - TCG - much better emulation framework
- 2010: I become linux-user maintainer
- 2014: 2.0: ARM64 target support
- 2016: 2.6: Much improved threading support

WHY?

- Bootstrapping new architectures (arm64,openrisc,j-core)
- Make Cross-Compiling easier
- Building container images for foreign architectures
- Running legacy closed source binaries

STATUS

- 30 targets supported, 9 hosts
- On amd64 host emulating arm64: LTP 58 Failures in 1158 tests
- Main broken usecase now: Java

HOW LINUX-USER WORKS

- Implements Linux kernel userspace interface
- System calls, ioctl, files in /proc
- Not unlike Windows Subsystem for Linux

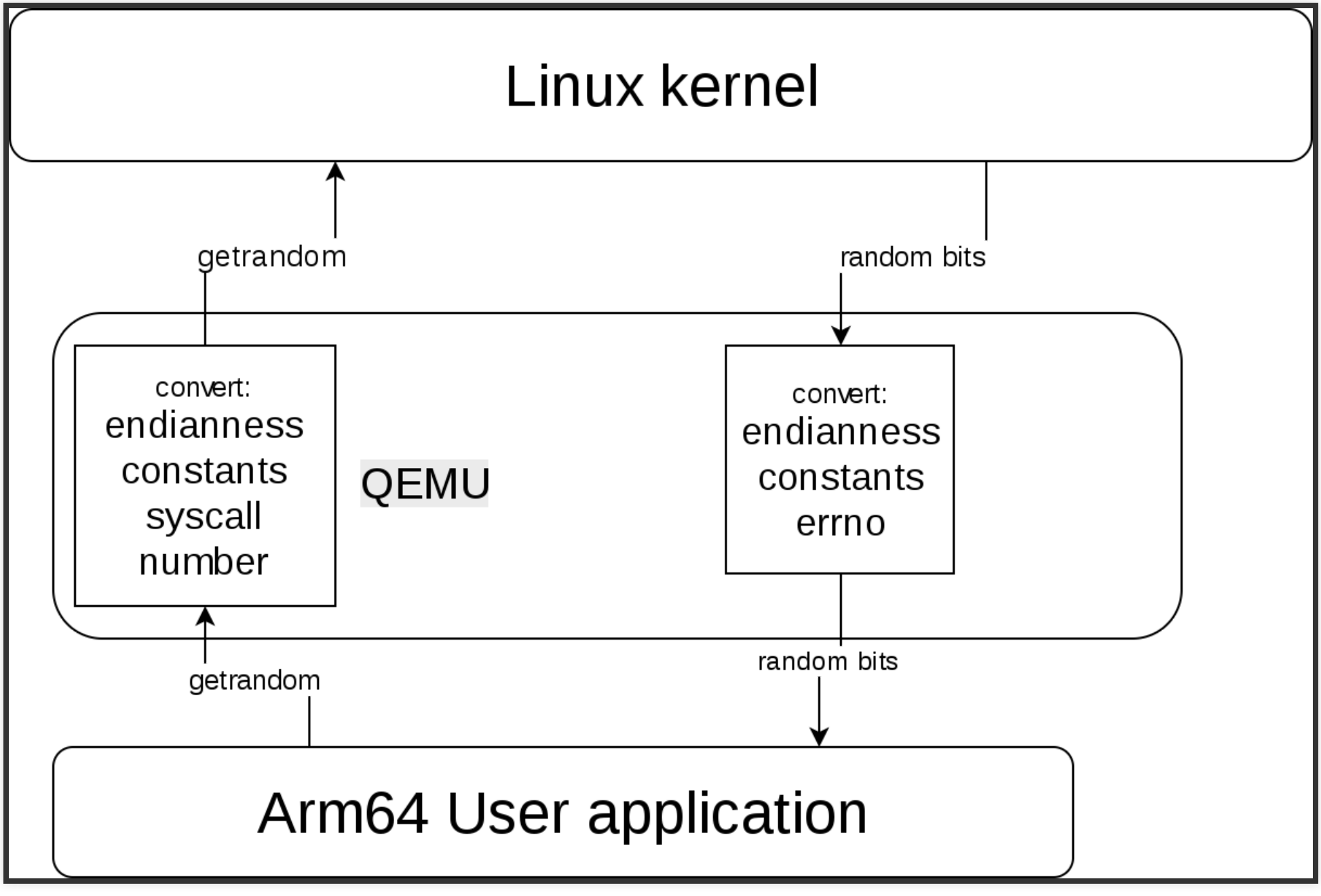
EXAMPLE

```
NAME
       getrandom - obtain a series of random bytes

SYNOPSIS
       #include <sys/random.h>

       int getrandom(void *buf, size_t buflen, unsigned int flags)

DESCRIPTION
       The  getrandom()  system call fills the buffer pointed to b
       with up to buflen random bytes.  These bytes can be used to
       user-space  random  number  generators or for cryptographic
       poses.
```


Linux kernel
getrandom
random bits
convert:
endianness
constants
syscall
number
QEMU
convert:
endianness
constants
errno
getrandom
random bits
Arm64 User application

QEMU SYSCALL.C:

```
#if defined(TARGET_NR_getrandom) && defined(__NR_getrandom)
        case TARGET_NR_getrandom:
        p = lock_user(VERIFY_WRITE, arg1, arg2, 0);
        if (!p) {
            goto efault;
        }
        ret = get_errno(getrandom(p, arg2, arg3));
        unlock_user(p, arg1, ret);
        break;
```

TYPICAL ISSUES

- Little/Big endian
- - get_user()/put_user() handle most cases
- - tswapX() when needed
- Structs and constant mapping
- - target_to_hostX() and vice versa
- Arch specific register/memory layouts

TESTING

- Instructions at https://wiki.qemu.org/Testing#User_mode_emulation
- Smoke testing tip: use static busybox
- Unit testing: Linux Testing Project aka LTP

SMOKETEST

Run static busybox of each architecture. Busybox has applets to test most common syscalls.

```
./qemu-aarch64 ./bin/busybox-arm64 ls -ld .
```

My static busybox collection

LTP

Linux testing project, tough testing of almost every syscall and linux userspace feature.

```
FROM arm64v8/debian:9

COPY qemu-aarch64 /usr/bin/qemu-aarch64-static

RUN apt-get update -q && \
    DEBIAN_FRONTEND=noninteractive apt-get install -q -y --no-inst
    build-essential xz-utils flex bison build-essential wget curl
    quota genisoimage sudo libaio-dev expect ca-certificates

RUN wget https://github.com/linux-test-project/ltp/releases/downlo
    tar xf ltp-full-20170516.tar.xz && \
    cd ltp-*; ./configure && \
    make -j8 && \
    make SKIP_IDCHECK=1 install
```

RUN LTP IN CONTAINER

```
docker run --privileged -v $(pwd)/aarch64-linux-user/qemu-aarch64:
/opt/ltp/runltp -p -l qemu.log -o qemu.out -f /opt/ltp/runtest/sys
https://lists.gnu.org/archive/html/qemu-devel/2016-10/msg04953.htm
```

FUTURE

- Easy place to join - lots of cleanups to do
- Try qemu user on other Hosts than X86 to find new bugs
- Examples: structure syscall.c, split signal.c
- Split code from #ifdef TARGET_X to target specific files

FUTURE #2

- Integrate smoketests and LTP into regular Qemu testing
- Follow the kernel and implement new syscalls
- Better support for android userland
- What about less maintained parts, BSD-user, unicore32, ..?
- If TCG becomes a library
- - Split linux-user to a indepedent project