Custom Linux system call

N.Mouzakitis

November 28, 2023

Contents

Abstract

Creation of a custom Linux system call in RISCV-64

1 Setup

In order to run using qemu-system-riscv64, a Linux environment and reate a new system call, Buildroot is used, in conjunction with OpenSBI. The modifications took place in the Buildroot sources and are described in the following section.

2 Modification in buildroot

2.1 output/build/linux-6.1.44/arch/riscv/kernel/syscall_table.c

```
1
2       // SPDX-License-Identifier: GPL-2.0-only
3       /*
4        * Copyright (C) 2009 Arnd Bergmann <arnd@arndb.de>
5        * Copyright (C) 2012 Regents of the University of California
6         */
7       #include <linux/linkage.h>
8       #include <linux/syscalls.h>
```

```
9       #include <asm-generic/syscalls.h>
10      #include <asm/syscall.h>
11
12      #undef __SYSCALL
13      #define __SYSCALL(nr, call) [nr] = (call),
14
15      void * const sys_call_table[__NR_syscalls] = {
16              [0 ... __NR_syscalls - 1] = sys_ni_syscall,
17        [__NR_custom_syscall] = sys_custom_syscall,
18      #include <asm/unistd.h>
19      };
```

Custom system call modification in line 17.

2.2 output/build/linux-6.1.44/arch/riscv/kernel/sys_riscv.c

```
1 // our definition of the custom syscall.
2 SYSCALL_DEFINE0(custom_syscall)
3 {
4       printk(KERN_ALERT "custom_syscall()\n");
5       return 0;  // Replace with actual return value
6 }
```

Addition in the file, defining and implementing the system call.

2.3 output/build/linux-6.1.44/include/uapi/asm-generic/unistd.h

```
1 #define __NR_custom_syscall 451
2 __SYSCALL(__NR_custom_syscall, sys_custom_syscall)
3
4 #undef __NR_syscalls
5 #define __NR_syscalls 452
```

Addition of syscall number 451, and update modification on the total number of system calls.

3 Usespace program in the guest, testing the new system call

```
1 #include <unistd.h>
2 #include <errno.h>
3 #include <sys/syscall.h>
4 #include <stdio.h>
5
6 #define SYS_custom_syscall 451
7
8 int main() {
9         //testing our custom syscall.
```

```
10      printf("testing custom syscall with NR: %d\n", SYS_custom_syscall);
11      long result = syscall(SYS_custom_syscall);
12
13      if (result == -1) {
14          perror("syscall error\n");
15      printf("errno: %ld \n", errno);
16          return 1;
17      }
18
19      printf("System call returned: %ld \n", result);
20
21      return 0;
22  }
```