SSE3052: Embedded Systems Practice

Jinkyu Jeong
jinkyu@skku.edu
Computer Systems Laboratory
Sungkyunkwan University
http://csl.skku.edu
Terminal

PC (Host)  Smartphone (Target)

adb (Android Debug Bridge)

• Command line tool that lets you communicate with an emulator instance or connected device.
  – Copying files to/from device.
  – Installing and debugging apps.
  – Running shell commands.
adb (Android Debug Bridge)

• Located in Android/Sdk/platform-tools.
  – Ex) ~/Android/Sdk/platform-tools

• Set PATH variable.
  – Ex) PATH=~/Android/Sdk/platform-tools:$PATH
  – (Append it to ~/.bashrc file for permanent change)
adb (Android Debug Bridge)

- Query for list of emulator/device instances.
  - `adb devices`

- Start a remote shell in the target instance.
  - `adb shell`
  - `adb -e shell`
  - `adb -d shell`
  - `adb -s <serialNumber> shell`
adb (Android Debug Bridge)

• Copy file to emulator/connected instance.
  – `adb push <local> <remote>`
  – Ex) `adb push foo.txt /data/local/tmp`

• Copy file from emulator/connected instance.
  – `adb pull <remote> <local>`

Agenda

1. Add a **system call** to Linux kernel.
2. Invoke added **system call** from user-level program.
Agenda

1. Add a **system call** to Linux kernel.
2. Invoke added **system call** from user-level program.
System Call?
main() function call foo() return

SSE3052: Embedded Systems Practice, Spring 2019, Jinkyu Jeong (jinkyu@skku.edu)
main() → function call → foo()

function_in_kernel() → function call → another_function_in_kernel()

User Space

Kernel Space

return

return
main() -> foo()

User Space

System Call Interface

Kernel Space

function_in_kernel() -> another_function_in_kernel()
Adding System Call

1. Add a new system call to the system call table.
2. Define (implement) the new system call.
3. Modify Makefile.
4. Build.

**Warning:** varies by kernel version & architecture.
(We use version: 3.10 & arch: x86_64)
Adding System Call

- In `goldfish/arch/x86/syscalls/syscall_64.tbl`
- Add a system call at the end of the entries.
Adding System Call

# The format is:
# <number> <abi> <name> <entry point>
#
# The abi is "common", "64" or "x32" for this file.

Ex) 322 common sunghwan_world sys_sunghwan_world
Adding System Call

• Create a new c file for implementing system call.
• Ex) goldfish/arch/x86/mm/sunghwan.c

```c
#include <linux/kernel.h>

asmlinkage long sys_sunghwan_world(void)
{
    printk("Hello, My name is Sunghwan!!!\n");
    return 0;
}
```
Adding System Call

- Modify Makefile.
- Ex) Add sunghwan.o to obj-y.

```makefile
# Kernel does not boot with instrumentation of tlb.c.
KCOV_INSTRUMENT_tlb.o := n

obj-y := init.o init_${BITS}.o fault.o ioremap.o extable.o pageattr.o mmap.o \\
        pat.o pgtable.o physaddr.o gup.o setup_ntx.o sunghwan.o

# Make sure __phys_addr has no stackprotector
nostackp := $(call cc-option, -fno-stack-protector)
CFLAGS_physaddr.o := $(nostackp)
CFLAGS_setup_ntx.o := $(nostackp)
CFLAGS_fault.o := -I$(src)/../include/asm/trace

obj-$(CONFIG_X86_PAT) += pat_rbtree.o
obj-$(CONFIG_SMP) += tlb.o
obj-$(CONFIG_X86_32) += pgtable_32.o iomap_32.o
obj-$(CONFIG_HUGETLB_PAGE) += hugetlbpage.o
obj-$(CONFIG_X86_PTDUMP) += dump_pagetables.o
obj-$(CONFIG_HIGHMEM) += highmem_32.o
```

"Makefile" 35L, 1063C
Adding System Call

• Build the kernel.
• Check out the system call number.
  – Ex) grep -nR sunghwan_world *

```bash
$ grep -nR sunghwan_world *
```

• Copy the kernel image to the appropriate directory.
Agenda

1. Add a **system call** to Linux kernel.

2. Invoke added **system call** from user-level program.
Invoking System Call

1. Set up toolchains. (cross compiler)
2. Write a user-level program that invokes newly added system call.
3. Compile.
4. Copy the executable to the device.
5. Execute.
6. Check out the message.
Invoking System Call

• Download NDK.
• Unzip.
• Execute the following:
  – [path_to_NDK]/build/tools/make_standalone_toolchain.py --arch x86_64 --api 24 --install-dir ~/my-android-toolchain
Invoking System Call

- Write a user-level program.
- Ex)

```
#include <unistd.h>
#define __NR_sunghwan_world 322

int main()
{
    syscall(__NR_sunghwan_world);
    return 0;
}
```
Invoking System Call

• Compile with \texttt{-pie} option.
  – Ex) \texttt{~/my-android-toolchain/bin/x86_64-linux-android-gcc -pie userspace.c}

• Copy to \texttt{/data/local/tmp} (target).
  – Ex) \texttt{adb push a.out /data/local/tmp}

• Execute & check out the message.
  – \texttt{su}
  – \texttt{dmesg}

```
[ 903.040506] ieee80211 phy0: mac80211_hwsim_config (freq=0 idle=0 ps=0 smps=auto)
[ 903.140209] ieee80211 phy0: mac80211_hwsim_config (freq=0 idle=0 ps=1 smps=auto)
[ 903.504794] healthd: battery l=100 v=5000 t=25.0 h=2 st=2 chg=a
[ 929.670569]   [ 934.080323] ieee80211 phy0: mac80211_hwsim_config (freq=0 idle=0 ps=0 smps=auto)
      generic_x86_64:/data/local/tmp #
```

\textbf{Hello, My name is Sunghwan!!!}
Lab

1. Add a system call that print own Student ID and NAME.
2. Invoke added system call from user-level program.
3. Submit report. (include Concept, Implementation, Result)

• Format: yourstudentID_lab1.pdf
  E-mail to: sunghwan.kim@csi.skku.edu
  Deadline: 3/20 (Wed) 23:59