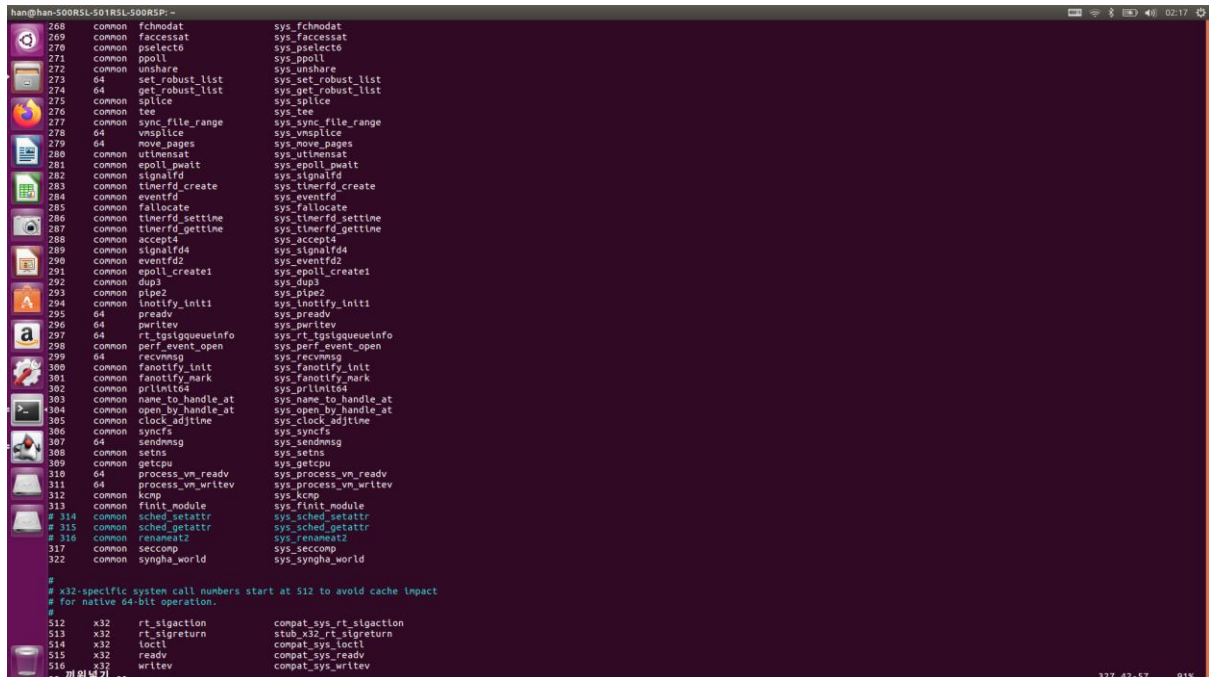


Embedded System Practice Lab 2

2016311821 한승하

2주차 실습 System call추가 보고서입니다.



우선 goldfish/arch/x86/syscalls/syscall_64.tbl 파일에 322 번 새로운 syscall 을 추가하여 주었습니다.

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

asmlinkage long sys_synga_world(void)
{
    printk("2016311821\nHan Synga\n");
    return 0;
}
```

이후 새로 추가한 syscall 의 handler 를 작성해 주었습니다. 학번과 이름이 출력될 수 있도록 작성해 주었습니다.

```
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_nx.o syscall_test.o

# Make sure __phys_addr has no stackprotector
nostackp := $(call cc-option, -fno-stack-protector)
CFLAGS_physaddr.o := $(nostackp)
CFLAGS_setup_nx.o := $(nostackp)

obj-$(CONFIG_X86_PAT) += pat_rbtrees.o
obj-$(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

obj-$(CONFIG_KMEMCHECK) += kmemcheck/

obj-$(CONFIG_MMIOTRACE) += mmioTRACE.o
mmioTRACE-y := kmio.o pf_in.o mmio-mod.o
obj-$(CONFIG_MMIOTRACE_TEST) += testmmioTRACE.o

obj-$(CONFIG_NUMA) += numa.o numa_${BITS}.o
obj-$(CONFIG_AMD_NUMA) += amdtopology.o
obj-$(CONFIG_ACPI_NUMA) += srat.o
obj-$(CONFIG_NUMA_EMU) += numa_emulation.o

obj-$(CONFIG_MEMTEST) += memtest.o
~
```

이후 Makefile 에 syscall_test.o 를 추가하여 컴파일 될 수 있도록 하였습니다.

```
han@han-500R5L-501R5L-500R5P:~$ vi goldfish/arch/x86/syscalls/syscall_64.tbl
han@han-500R5L-501R5L-500R5P:~$ vi goldfish/arch/x86/mm/syscall_test.c
han@han-500R5L-501R5L-500R5P:~$ vi goldfish/arch/x86/mm/Makefile
han@han-500R5L-501R5L-500R5P:~$ export ARCH=x86_64
han@han-500R5L-501R5L-500R5P:~$ export CROSS_COMPILE=~/.x86_64-linux-android-4.9/bin/x86_64-linux-android-
han@han-500R5L-501R5L-500R5P:~$ make x86_64_ranchu_defconfig
make: *** 타겟 'x86_64_ranchu_defconfig'을(를) 만들 규칙이 없습니다. 멈춤.
han@han-500R5L-501R5L-500R5P:~$ cd goldfish/
han@han-500R5L-501R5L-500R5P:~/goldfish$ make x86_64_ranchu_defconfig
arch/x86/configs/x86_64_ranchu_defconfig:128:warning: override: reassigning to symbol NETFILTER_TPROXY
#
# configuration written to .config
#
han@han-500R5L-501R5L-500R5P:~/goldfish$ make -j
scripts/kconfig/conf --silentoldconfig Kconfig
^C
han@han-500R5L-501R5L-500R5P:~/goldfish$ make -j4
SYSHDR arch/x86/syscalls/./include/generated/uapi/asm/unistd_64.h
SYSHDR arch/x86/syscalls/./include/generated/uapi/asm/unistd_x32.h
SYSHDR arch/x86/syscalls/./include/generated/asm/unistd_64_x32.h
SYSTBL arch/x86/syscalls/./include/generated/asm/syscalls_64.h
CHK include/generated/uapi/linux/version.h
CHK include/generated/utsrelease.h
CC scripts/mod/devicetable-offsets.s
GEN scripts/mod/devicetable-offsets.h
HOSTCC scripts/mod/file2alias.o
make[1]: 'relocs'을(를) 위해 할 일이 없습니다.
HOSTLD scripts/mod/modpost
CC arch/x86/kernel/asm-offsets.s
GEN include/generated/asm-offsets.h
```

이후 이전 실습시간에 진행한 대로 컴파일 하여 bzImage 를 생성해 주었고,

```
Kernel: arch/x86/boot/bzImage is ready (#6)
han@han-500R5L-501R5L-500R5P:~/goldfish$ cd ~/Android/Sdk/system-images/android-24/google_apis/x86_64
han@han-500R5L-501R5L-500R5P:~/Android/Sdk/system-images/android-24/google_apis/x86_64$ mv kernel-ranchu kernel-ranchu.bak
han@han-500R5L-501R5L-500R5P:~/Android/Sdk/system-images/android-24/google_apis/x86_64$ cp ~/goldfish/arch/x86/b
boot/ built-in.o
han@han-500R5L-501R5L-500R5P:~/Android/Sdk/system-images/android-24/google_apis/x86_64$ cp ~/goldfish/arch/x86/boot/bzImage kernel-ranchu
han@han-500R5L-501R5L-500R5P:~/Android/Sdk/system-images/android-24/google_apis/x86_64$
```

생성된 이미지를 실행할 에뮬레이터 폴더에 복사하여 주었습니다.

```
han@han-500R5L-501R5L-500R5P:~/goldfish$ grep -nR syngha_world *
System.map:1449:ffffffff81027dc2 T sys_syngha_world
이진파일 arch/x86/mm/built-in.o 와(과) 일치
arch/x86/mm/syscall_test.c:3:asmlinkage long sys_syngha_world(void)
이진파일 arch/x86/mm/syscall_test.o 와(과) 일치
이진파일 arch/x86/built-in.o 와(과) 일치
이진파일 arch/x86/kernel/built-in.o 와(과) 일치
이진파일 arch/x86/kernel/syscall_64.o 와(과) 일치
arch/x86/syscalls/syscall_64.tbl:327:322          common syngha_world          sys_syngha_world
arch/x86/include/generated/uapi/asm/unistd_64.h:319:#define __NR_syngha_world 322
arch/x86/include/generated/uapi/asm/unistd_x32.h:277:#define __NR_syngha_world (__X32_SYSCALL_BIT + 322)
arch/x86/include/generated/asm/syscalls_64.h:301:__SYSCALL_COMMON(322, sys_syngha_world, sys_syngha_world)
이진파일 vmlinux 와(과) 일치
이진파일 vmlinux.o 와(과) 일치
```

Grep 을 사용하여 확인한 결과 정상적으로 syscall 이 추가되었음을 확인할 수 있었습니다.

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

int main()
{
    syscall(__NR_syngha_world);
    return 0;
}
```

이후 에뮬레이터에서 실행시킬 실행파일을 작성해 주었습니다.

```
han@han-500R5L-501R5L-500R5P:~$ ~/my-android-toolchain/bin/x86_64-linux-android-gcc -pie userspace.c
han@han-500R5L-501R5L-500R5P:~$ rm a.out
han@han-500R5L-501R5L-500R5P:~$ ls
Android AndroidStudioProjects android-studio examples.desktop goldfish my-android-toolchain user
han@han-500R5L-501R5L-500R5P:~$ ~/my-android-toolchain/bin/x86_64-linux-android-gcc -pie userspace.c
han@han-500R5L-501R5L-500R5P:~$ ls
Android AndroidStudioProjects a.out android-studio examples.desktop goldfish my-android-toolchain
han@han-500R5L-501R5L-500R5P:~$
```

작성한 실행파일을 컴파일하여 실행파일 a.out 을 만들어 주었습니다.

```
[ 95.643350] 2016311821
[ 95.643350] Han Syngha
generic_x86_64:/data/local/tmp # dmesg | grep "2016311821"
[ 95.643350] 2016311821
generic_x86_64:/data/local/tmp # dmesg | grep "Han"
[ 95.643350] Han Syngha
generic_x86_64:/data/local/tmp #
```

만든 파일을 adb push 이후 실행시켜 확인해본 결과 정상적으로 출력되는 것을 확인할 수 있었습니다. Syscall 출력에 \n 으로 줄을 나누어서 2 줄에 걸쳐 출력되는 것을 확인하였습니다.