浙江大学实验报告

课程名称：操作系统　　实验类型：综合型

实验项目名称：添加系统调用

学生姓名：张佳瑶　　学号：3170103240

电子邮件地址：1531077171@qq.com

实验日期：2019 年 12 月 13 日

一、实验环境

处理器：Intel® Core™ i7-6700HQ CPU @ 2.60GHz

Windows10

Linux version 5.0.0-29-generic (buildd@lgw01-amd64-039) (gcc version 7.4.0 (Ubuntu 7.4.0-1ubuntu1~18.04.1))

二、实验内容和结果及分析

查看当前 Ubuntu 版本：

```
zjy@ubuntu:~$ cat /proc/version
Linux version 4.15.0-45-generic (buildd@lcy01-amd64-027) (gcc version 5.4.0 2016
0609 (Ubuntu 5.4.0-6ubuntu1~16.04.10)) #48~16.04.1-Ubuntu SMP Tue Jan 29 18:03:4
8 UTC 2019
```

查看内核源代码：

```
zjy@ubuntu:~$ sudo apt-cache search linux-source
[sudo] password for zjy:
linux-source - Linux kernel source with Ubuntu patches
linux-source-4.4.0 - Linux kernel source for version 4.4.0 with Ubuntu patches
linux-source-4.10.0 - Linux kernel source for version 4.10.0 with Ubuntu patches
linux-source-4.11.0 - Linux kernel source for version 4.11.0 with Ubuntu patches
linux-source-4.13.0 - Linux kernel source for version 4.13.0 with Ubuntu patches
linux-source-4.15.0 - Linux kernel source for version 4.15.0 with Ubuntu patches
linux-source-4.8.0 - Linux kernel source for version 4.8.0 with Ubuntu patches
```

当前 Ubuntu 版本为 Ubuntu16.04.10，内核版本为 4.15.0。

准备编译内核，下载 4.15.0 版本的内核源码：

```
sudo apt-get install linux-source-4.15.0
```

```
zjy@ubuntu:~$ sudo apt-get install linux-source-4.15.0
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  libncurses-dev | ncurses-dev kernel-package libqt3-dev
The following NEW packages will be installed:
  linux-source-4.15.0
0 upgraded, 1 newly installed, 0 to remove and 306 not upgraded.
Need to get 129 MB of archives.
After this operation, 146 MB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 linux-source-
4.15.0 all 4.15.0-72.81~16.04.1 [129 MB]
0% [1 linux-source-4.15.0 51.8 kB/129 MB 0%]
```

下载好的内核源代码默认放置在/usr/src：

```
cd /usr/src
ls
```

```
zjy@ubuntu:~$ cd  /usr/src
zjy@ubuntu:/usr/src$ ls
linux-headers-4.15.0-45          linux-source-4.8.0
linux-headers-4.15.0-45-generic  linux-source-4.8.0.tar.bz2
```

将其拷贝到新建的文件夹 build_kernel 下，并解压：

```
cd
mkdir build_kernel
cd build_kernel
cp /usr/src/linux-source-4.15.0.tar.bz2 .
tar jxvf linux-source-4.15.0.tar.bz2
```

结果为：

```
linux-source-4.15.0/block/partitions/atari.h
linux-source-4.15.0/block/partitions/amiga.h
linux-source-4.15.0/block/partitions/aix.c
linux-source-4.15.0/block/partitions/karma.c
linux-source-4.15.0/block/partitions/ultrix.h
linux-source-4.15.0/block/partitions/sgi.h
linux-source-4.15.0/block/partitions/mac.c
linux-source-4.15.0/block/partitions/check.c
linux-source-4.15.0/block/blk-exec.c
linux-source-4.15.0/block/scsi_ioctl.c
linux-source-4.15.0/block/blk-integrity.c
linux-source-4.15.0/block/bio.c
linux-source-4.15.0/block/blk-mq-debugfs.c
linux-source-4.15.0/block/blk-ioc.c
linux-source-4.15.0/block/blk-wbt.h
linux-source-4.15.0/block/Makefile
linux-source-4.15.0/block/cfq-iosched.c
linux-source-4.15.0/block/kyber-iosched.c
linux-source-4.15.0/block/Kconfig.iosched
linux-source-4.15.0/block/blk.h
linux-source-4.15.0/block/blk-lib.c
linux-source-4.15.0/block/bfq-wf2q.c
linux-source-4.15.0/block/blk-mq-tag.h
linux-source-4.15.0/block/blk-merge.c
zjy@ubuntu:~/build_kernel$ ls
linux-source-4.15.0  linux-source-4.15.0.tar.bz2
```

下载库：

```
sudo apt-get install libncurses5-dev libssl-dev
```

```
zjy@ubuntu:~/build_kernel/linux-source-4.15.0$ sudo apt-get install libncurses5-dev libssl-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libssl-doc libssl1.0.0 libtinfo-dev zlib1g-dev
Suggested packages:
  ncurses-doc
The following NEW packages will be installed:
  libncurses5-dev libssl-dev libssl-doc libtinfo-dev zlib1g-dev
The following packages will be upgraded:
  libssl1.0.0
1 upgraded, 5 newly installed, 0 to remove and 305 not upgraded.
Need to get 3,925 kB of archives.
After this operation, 12.0 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libssl1.0.0 amd64 1.0.2g-1ubuntu4.15 [1,084 kB]
Get:2 http://us.archive.ubuntu.com/ubuntu xenial/main amd64 libtinfo-dev amd64 6.0+20160213-1ubuntu1 [77.4 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu xenial/main amd64 libncurses5-dev amd64 6.0+20160213-1ubuntu1 [175 kB]
Get:4 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 zlib1g-dev amd64 1:1.2.8.dfsg-2ubuntu4.1 [168 kB]
Get:5 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libssl-dev amd64 1.0.2g-1ubuntu4.15 [1,344 kB]
Get:6 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libssl-doc all 1.0.2g-1ubuntu4.15 [1,077 kB]
Fetched 3,925 kB in 34s (115 kB/s)
Preconfiguring packages ...
(Reading database ... 178280 files and directories currently installed.)
Preparing to unpack .../libssl1.0.0_1.0.2g-1ubuntu4.15_amd64.deb ...
Unpacking libssl1.0.0:amd64 (1.0.2g-1ubuntu4.15) over (1.0.2g-1ubuntu4.14) ...
Selecting previously unselected package libtinfo-dev:amd64.
Preparing to unpack .../libtinfo-dev_6.0+20160213-1ubuntu1_amd64.deb ...
Unpacking libtinfo-dev:amd64 (6.0+20160213-1ubuntu1) ...
Selecting previously unselected package libncurses5-dev:amd64.
Preparing to unpack .../libncurses5-dev_6.0+20160213-1ubuntu1_amd64.deb ...
Unpacking libncurses5-dev:amd64 (6.0+20160213-1ubuntu1) ...
Selecting previously unselected package zlib1g-dev:amd64.
Preparing to unpack .../zlib1g-dev_1%3a1.2.8.dfsg-2ubuntu4.1_amd64.deb ...
Unpacking zlib1g-dev:amd64 (1:1.2.8.dfsg-2ubuntu4.1) ...
Selecting previously unselected package libssl-dev:amd64.
Preparing to unpack .../libssl-dev_1.0.2g-1ubuntu4.15_amd64.deb ...
Unpacking libssl-dev:amd64 (1.0.2g-1ubuntu4.15) ...
Selecting previously unselected package libssl-doc.
Preparing to unpack .../libssl-doc_1.0.2g-1ubuntu4.15_all.deb ...
Unpacking libssl-doc (1.0.2g-1ubuntu4.15) ...
```

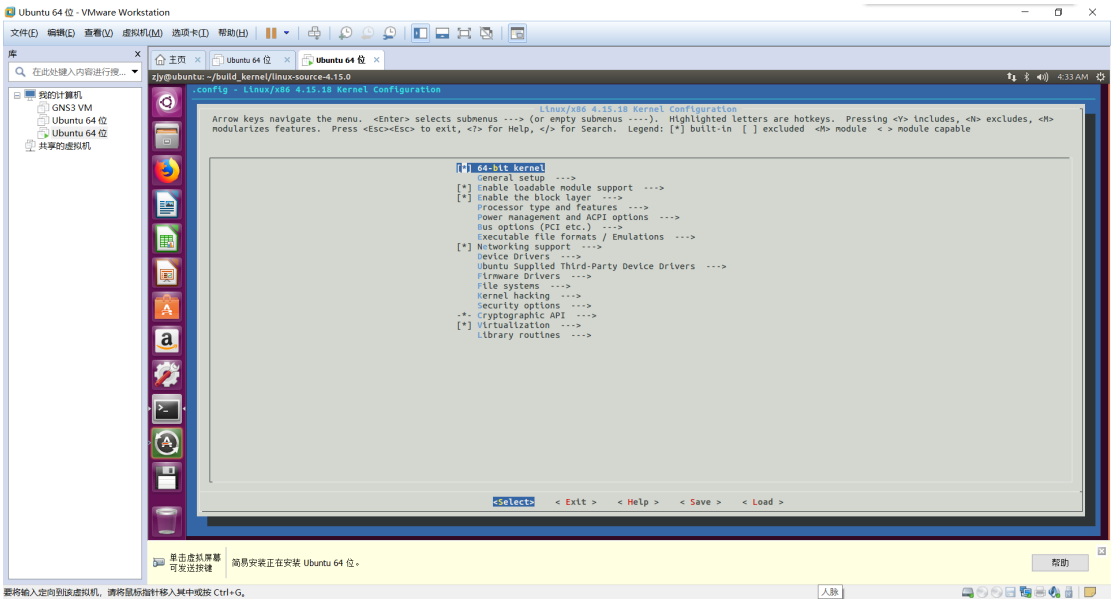
配置内核：

```
zjy@ubuntu:~/build_kernel$ cp /usr/src/linux-headers-4.15.0-45-generic/.config ./linux-source-4.15.0
zjy@ubuntu:~/build_kernel$ cd linux-source-4.15.0
zjy@ubuntu:~/build_kernel/linux-source-4.15.0$ make menuconfig
```

```
zjy@ubuntu:~/build_kernel$ cp /usr/src/linux-headers-4.15.0-45-generic/.config .
```

```
zjy@ubuntu:~/build_kernel/linux-source-4.15.0$ make menuconfig
  HOSTCC  scripts/kconfig/mconf.o
  SHIPPED scripts/kconfig/zconf.tab.c
  SHIPPED scripts/kconfig/zconf.lex.c
  HOSTCC  scripts/kconfig/zconf.tab.o
  HOSTCC  scripts/kconfig/lxdialog/checklist.o
  HOSTCC  scripts/kconfig/lxdialog/util.o
  HOSTCC  scripts/kconfig/lxdialog/inputbox.o
  HOSTCC  scripts/kconfig/lxdialog/textbox.o
  HOSTCC  scripts/kconfig/lxdialog/yesno.o
  HOSTCC  scripts/kconfig/lxdialog/menubox.o
  HOSTLD  scripts/kconfig/mconf
scripts/kconfig/mconf  Kconfig
```

make menuconfig 菜单出现后依次选择 load，OK，save，OK，exit，exit：




```
zjy@ubuntu:~/build_kernel/linux-source-4.15.0$ make menuconfig
  HOSTCC  scripts/kconfig/mconf.o
  SHIPPED scripts/kconfig/zconf.tab.c
  SHIPPED scripts/kconfig/zconf.lex.c
  HOSTCC  scripts/kconfig/zconf.tab.o
  HOSTCC  scripts/kconfig/lxdialog/checklist.o
  HOSTCC  scripts/kconfig/lxdialog/util.o
  HOSTCC  scripts/kconfig/lxdialog/inputbox.o
  HOSTCC  scripts/kconfig/lxdialog/textbox.o
  HOSTCC  scripts/kconfig/lxdialog/yesno.o
  HOSTCC  scripts/kconfig/lxdialog/menubox.o
  HOSTLD  scripts/kconfig/mconf
scripts/kconfig/mconf  Kconfig
security/Kconfig:393:warning: defaults for choice values not supported
security/Kconfig:397:warning: defaults for choice values not supported
security/Kconfig:401:warning: defaults for choice values not supported
security/Kconfig:405:warning: defaults for choice values not supported
security/Kconfig:409:warning: defaults for choice values not supported


*** End of the configuration.
*** Execute 'make' to start the build or try 'make help'.
```

内核编译准备结束，开始第一次编译内核：

```
make -j8
```

```
  LD [M]  sound/soundcore.ko
  LD [M]  sound/synth/emux/snd-emux-synth.ko
  LD [M]  sound/synth/snd-util-mem.ko
  LD [M]  sound/usb/6fire/snd-usb-6fire.ko
  LD [M]  sound/usb/bcd2000/snd-bcd2000.ko
  LD [M]  sound/usb/caiaq/snd-usb-caiaq.ko
  LD [M]  sound/usb/hiface/snd-usb-hiface.ko
  LD [M]  sound/usb/line6/snd-usb-line6.ko
  LD [M]  sound/usb/line6/snd-usb-pod.ko
  LD [M]  sound/usb/line6/snd-usb-podhd.ko
  LD [M]  sound/usb/line6/snd-usb-toneport.ko
  LD [M]  sound/usb/line6/snd-usb-variax.ko
  LD [M]  sound/usb/misc/snd-ua101.ko
  LD [M]  sound/usb/snd-usb-audio.ko
  LD [M]  sound/usb/snd-usbmidi-lib.ko
  LD [M]  sound/usb/usx2y/snd-usb-us122l.ko
  LD [M]  sound/usb/usx2y/snd-usb-usx2y.ko
  LD [M]  sound/x86/snd-hdmi-lpe-audio.ko
  LD [M]  ubuntu/hio/hio.ko
  LD [M]  ubuntu/vbox/vboxguest/vboxguest.ko
  LD [M]  ubuntu/vbox/vboxsf/vboxsf.ko
  LD [M]  ubuntu/xr-usb-serial/xr_usb_serial_common.ko
  LD [M]  virt/lib/irqbypass.ko
zjy@ubuntu:~/build_kernel/linux-source-4.15.0$
```

sudo make modules_install -j 8

```
  INSTALL sound/synth/emux/snd-emux-synth.ko
  INSTALL sound/synth/snd-util-mem.ko
  INSTALL sound/usb/bcd2000/snd-bcd2000.ko
  INSTALL sound/usb/6fire/snd-usb-6fire.ko
  INSTALL sound/usb/caiaq/snd-usb-caiaq.ko
  INSTALL sound/usb/hiface/snd-usb-hiface.ko
  INSTALL sound/usb/line6/snd-usb-line6.ko
  INSTALL sound/usb/line6/snd-usb-pod.ko
  INSTALL sound/usb/line6/snd-usb-podhd.ko
  INSTALL sound/usb/line6/snd-usb-toneport.ko
  INSTALL sound/usb/line6/snd-usb-variax.ko
  INSTALL sound/usb/snd-usb-audio.ko
  INSTALL sound/usb/misc/snd-ua101.ko
  INSTALL sound/usb/snd-usbmidi-lib.ko
  INSTALL sound/usb/usx2y/snd-usb-us122l.ko
  INSTALL sound/usb/usx2y/snd-usb-usx2y.ko
  INSTALL sound/x86/snd-hdmi-lpe-audio.ko
  INSTALL ubuntu/hio/hio.ko
  INSTALL ubuntu/vbox/vboxguest/vboxguest.ko
  INSTALL ubuntu/vbox/vboxsf/vboxsf.ko
  INSTALL ubuntu/xr-usb-serial/xr_usb_serial_common.ko
  INSTALL virt/lib/irqbypass.ko
  DEPMOD  4.15.18
zjy@ubuntu:~/build_kernel/linux-source-4.15.0$
```

sudo make install -j 8

```
run-parts: executing /etc/kernel/postinst.d/apt-auto-removal 4.15.18 /boot/vmlin
uz-4.15.18
run-parts: executing /etc/kernel/postinst.d/initramfs-tools 4.15.18 /boot/vmlinu
z-4.15.18
update-initramfs: Generating /boot/initrd.img-4.15.18
run-parts: executing /etc/kernel/postinst.d/pm-utils 4.15.18 /boot/vmlinuz-4.15.
18
run-parts: executing /etc/kernel/postinst.d/unattended-upgrades 4.15.18 /boot/vm
linuz-4.15.18
run-parts: executing /etc/kernel/postinst.d/update-notifier 4.15.18 /boot/vmlinu
z-4.15.18
run-parts: executing /etc/kernel/postinst.d/zz-update-grub 4.15.18 /boot/vmlinuz
-4.15.18
Generating grub configuration file ...
Warning: Setting GRUB_TIMEOUT to a non-zero value when GRUB_HIDDEN_TIMEOUT is se
t is no longer supported.
Found linux image: /boot/vmlinuz-4.15.18
Found initrd image: /boot/initrd.img-4.15.18
Found linux image: /boot/vmlinuz-4.15.0-45-generic
Found initrd image: /boot/initrd.img-4.15.0-45-generic
Found memtest86+ image: /boot/memtest86+.elf
Found memtest86+ image: /boot/memtest86+.bin
done
zjy@ubuntu:~/build_kernel/linux-source-4.15.0$
```

sudo update-initramfs -c -k 4.15.18

```
zjy@ubuntu:~/build_kernel/linux-source-4.15.0$ sudo update-initramfs -c -k 4.15.
18
update-initramfs: Generating /boot/initrd.img-4.15.18
```

sudo update-grub

```
zjy@ubuntu:~/build_kernel/linux-source-4.15.0$ sudo update-grub
Generating grub configuration file ...
Warning: Setting GRUB_TIMEOUT to a non-zero value when GRUB_HIDDEN_TIMEOUT is se
t is no longer supported.
Found linux image: /boot/vmlinuz-4.15.18
Found initrd image: /boot/initrd.img-4.15.18
Found linux image: /boot/vmlinuz-4.15.0-45-generic
Found initrd image: /boot/initrd.img-4.15.0-45-generic
Found memtest86+ image: /boot/memtest86+.elf
Found memtest86+ image: /boot/memtest86+.bin
done
```

重启，验证内核是否编译成功

```
zjy@ubuntu:~$ uname -a
Linux ubuntu 4.15.18 #1 SMP Thu Dec 12 06:26:09 PST 2019 x86_64 x86_64 x86_64 GN
U/Linux
```

内核版本已经更新为 4.15.18，证明内核编译成功。

开始准备添加系统调用。

以下统一在下载的内核源码 ~/build*kernel/linux-source-4.15.0 文件夹下，在 arch/x86/entry/syscalls/syscall*64.tbl 添加 `333 common mysyscall sys_mysyscall`。

```
302     common  prlimit64               sys_prlimit64
303     common  name_to_handle_at       sys_name_to_handle_at
304     common  open_by_handle_at       sys_open_by_handle_at
305     common  clock_adjtime           sys_clock_adjtime
306     common  syncfs                  sys_syncfs
307     64      sendmmsg                sys_sendmmsg
308     common  setns                   sys_setns
309     common  getcpu                  sys_getcpu
310     64      process_vm_readv        sys_process_vm_readv
311     64      process_vm_writev       sys_process_vm_writev
312     common  kcmp                    sys_kcmp
313     common  finit_module            sys_finit_module
314     common  sched_setattr           sys_sched_setattr
315     common  sched_getattr           sys_sched_getattr
316     common  renameat2               sys_renameat2
317     common  seccomp                 sys_seccomp
318     common  getrandom               sys_getrandom
319     common  memfd_create            sys_memfd_create
320     common  kexec_file_load         sys_kexec_file_load
321     common  bpf                     sys_bpf
322     64      execveat                sys_execveat/ptregs
323     common  userfaultfd             sys_userfaultfd
324     common  membarrier              sys_membarrier
325     common  mlock2                  sys_mlock2
326     common  copy_file_range         sys_copy_file_range
327     64      preadv2                 sys_preadv2
328     64      pwritev2                sys_pwritev2
329     common  pkey_mprotect           sys_pkey_mprotect
330     common  pkey_alloc              sys_pkey_alloc
331     common  pkey_free               sys_pkey_free
332     common  statx                   sys_statx
333     common  mysyscall               sys_mysyscall

#
# x32-specific system call numbers start at 512 to avoid cache impact
# for native 64-bit operation.
#
                                        Plain Text ▾  Tab Width: 8 ▾   Ln 342, Col 54   ▾   INS
```

在 include/linux/mm.h 中添加 `extern unsigned pfcount long pfcount;`。

*mm.h (~/build_kernel/linux-source-4.15.0/include/linux) - gedit

Open Save

```
/* SPDX-License-Identifier: GPL-2.0 */
#ifndef _LINUX_MM_H
#define _LINUX_MM_H

extern unsigned long pfcount;

#include <linux/errno.h>

#ifdef __KERNEL__

#include <linux/mmdebug.h>
#include <linux/gfp.h>
#include <linux/bug.h>
#include <linux/list.h>
#include <linux/mmzone.h>
#include <linux/rbtree.h>
#include <linux/atomic.h>
#include <linux/debug_locks.h>
#include <linux/mm_types.h>
#include <linux/range.h>
#include <linux/pfn.h>
#include <linux/percpu-refcount.h>
#include <linux/bit_spinlock.h>
#include <linux/shrinker.h>
#include <linux/resource.h>
#include <linux/page_ext.h>
#include <linux/err.h>
#include <linux/page_ref.h>
#include <linux/memremap.h>

struct mempolicy;
struct anon_vma;
struct anon_vma_chain;
struct file_ra_state;
struct user_struct;
struct writeback_control;
struct bdi_writeback;
```

C/C++/ObjC Header Tab Width: 8 Ln 6, Col 1 INS

在 include/linux/sched.h 中添加 `unsigned long pf;`

sched.h (~/build_kernel/linux-source-4.15.0/include/linux) - gedit

Open Save

```
        u64                             gtime;
        struct prev_cputime             prev_cputime;
#ifdef CONFIG_VIRT_CPU_ACCOUNTING_GEN
        struct vtime                    vtime;
#endif

#ifdef CONFIG_NO_HZ_FULL
        atomic_t                        tick_dep_mask;
#endif
        /* Context switch counts: */
        unsigned long                   nvcsw;
        unsigned long                   nivcsw;

        unsigned long                   pf;

        /* Monotonic time in nsecs: */
        u64                             start_time;

        /* Boot based time in nsecs: */
        u64                             real_start_time;

        /* MM fault and swap info: this can arguably be seen as either mm-specific or thread-
specific: */
        unsigned long                   min_flt;
        unsigned long                   maj_flt;

#ifdef CONFIG_POSIX_TIMERS
        struct task_cputime             cputime_expires;
        struct list_head                cpu_timers[3];
#endif

        /* Process credentials: */

        /* Tracer's credentials at attach: */
        const struct cred __rcu         *ptracer_cred;

        /* Objective and real subjective task credentials (COW): */
```

C/C++/ObjC Header Tab Width: 8 Ln 769, Col 1 INS

在 kernel/fork.c 中添加 `tsk->pf = 0;`

fork.c (~/build_kernel/linux-source-4.15.0/kernel) - gedit

Open Save

```
        stackend = end_of_stack(tsk);
        *stackend = STACK_END_MAGIC;    /* for overflow detection */
}

static struct task_struct *dup_task_struct(struct task_struct *orig, int node)
{
        struct task_struct *tsk;
        unsigned long *stack;
        struct vm_struct *stack_vm_area;
        int err;

        if (node == NUMA_NO_NODE)
                node = tsk_fork_get_node(orig);
        tsk = alloc_task_struct_node(node);
        if (!tsk)
                return NULL;

        stack = alloc_thread_stack_node(tsk, node);
        if (!stack)
                goto free_tsk;

        stack_vm_area = task_stack_vm_area(tsk);

        err = arch_dup_task_struct(tsk, orig);

        tsk->pf = 0;

        /*
         * arch_dup_task_struct() clobbers the stack-related fields.  Make
         * sure they're properly initialized before using any stack-related
         * functions again.
         */
        tsk->stack = stack;
#ifdef CONFIG_VMAP_STACK
        tsk->stack_vm_area = stack_vm_area;
#endif
```

C Tab Width: 8 Ln 563, Col 1 INS

在 arch/x86/mm/fault.c 中添加 unsigned long pfcount; 和 pfcount++;current->pf++;

fault.c (~/build_kernel/linux-source-4.15.0/arch/x86/mm) - gedit

Open Save

```c
NOKPROBE_SYMBOL(do_user_addr_fault);

/*
 * This routine handles page faults.  It determines the address,
 * and the problem, and then passes it off to one of the appropriate
 * routines.
 */
unsigned long pfcount;
static noinline void
__do_page_fault(struct pt_regs *regs, unsigned long hw_error_code,
                unsigned long address)
{
        prefetchw(&current->mm->mmap_sem);

        pfcount++;
        current->pf++;

        if (unlikely(kmmio_fault(regs, address)))
                return;

        /* Was the fault on kernel-controlled part of the address space? */
        if (unlikely(fault_in_kernel_space(address)))
                do_kern_addr_fault(regs, hw_error_code, address);
        else
                do_user_addr_fault(regs, hw_error_code, address);
}
NOKPROBE_SYMBOL(__do_page_fault);

static nokprobe_inline void
trace_page_fault_entries(unsigned long address, struct pt_regs *regs,
                         unsigned long error_code)
{
        if (user_mode(regs))
                trace_page_fault_user(address, regs, error_code);
        else
                trace_page_fault_kernel(address, regs, error_code);
}
```

C Tab Width: 8 Ln 1499, Col 43 INS

在 kernel/sys.c 中添加

```
SYSCALL_DEFINE0(mysyscall)
```

build_kernel/linux-source-4.15.0/kernel) - gedit

Open

```
 *
 * This is SMP safe as current->tgid does not change.
 */
SYSCALL_DEFINE0(getpid)
{
        return task_tgid_vnr(current);
}

/* Thread ID - the internal kernel "pid" */
SYSCALL_DEFINE0(gettid)
{
        return task_pid_vnr(current);
}

SYSCALL_DEFINE0(mysyscall)
{
        printk("system process - page fault count %ld \n", pfcount);
        printk("current process - page fault count %ld \n", current->pf);
        printk("dirty page of all processes:\n");
        struct task_struct *p = NULL;
        for(p = &init_task; (p = next_task(p)) != &init_task; )
        {
                printk("pid:%ld--dirty page:%d\n",p->pid,p->nr_dirtied);
        }
        return pfcount;
}

/*
 * Accessing ->real_parent is not SMP-safe, it could
 * change from under us. However, we can use a stale
 * value of ->real_parent under rcu_read_lock(), see
 * release_task()->call_rcu(delayed_put_task_struct).
 */
SYSCALL_DEFINE0(getppid)
{
        int pid;

        rcu_read_lock();
        pid = task_tgid_vnr(rcu_dereference(current->real_parent));
        rcu_read_unlock();

        return pid;
}

SYSCALL_DEFINE0(getuid)
{
        /* Only we change this so SMP safe */
```

然后重新编译内核：

```
make -j8
sudo make modules_install -j 8
sudo make install -j 8
sudo update-initramfs -c -k 4.8.15
sudo update-grub
```

```
zjy@ubuntu: ~/build_kernel/linux-source-4.15.0
  LD [M]  sound/soc/zte/zx-tdm.ko
  LD [M]  sound/soundcore.ko
  LD [M]  sound/synth/emux/snd-emux-synth.ko
  LD [M]  sound/synth/snd-util-mem.ko
  LD [M]  sound/usb/6fire/snd-usb-6fire.ko
  LD [M]  sound/usb/bcd2000/snd-bcd2000.ko
  LD [M]  sound/usb/caiaq/snd-usb-caiaq.ko
  LD [M]  sound/usb/hiface/snd-usb-hiface.ko
  LD [M]  sound/usb/line6/snd-usb-line6.ko
  LD [M]  sound/usb/line6/snd-usb-pod.ko
  LD [M]  sound/usb/line6/snd-usb-podhd.ko
  LD [M]  sound/usb/line6/snd-usb-toneport.ko
  LD [M]  sound/usb/line6/snd-usb-variax.ko
  LD [M]  sound/usb/misc/snd-ua101.ko
  LD [M]  sound/usb/snd-usb-audio.ko
  LD [M]  sound/usb/snd-usbmidi-lib.ko
  LD [M]  sound/usb/usx2y/snd-usb-us122l.ko
  LD [M]  sound/usb/usx2y/snd-usb-usx2y.ko
  LD [M]  sound/x86/snd-hdmi-lpe-audio.ko
  LD [M]  ubuntu/hio/hio.ko
  LD [M]  ubuntu/vbox/vboxguest/vboxguest.ko
  LD [M]  ubuntu/vbox/vboxsf/vboxsf.ko
  LD [M]  ubuntu/xr-usb-serial/xr_usb_serial_common.ko
zjy@ubuntu:~/build_kernel/linux-source-4.15.0$
```

重启。

编写用户态程序并编译。

```
#include <linux/unistd.h>
#include <sys/syscall.h>
#define __NR_mysyscall 333

int main()
{
    pfcount = syscall(__NR_mysyscall);

}
```

编译用户态程序并运行，然后输入 dmesg 验证：

```
[  112.266136] system process - page fault count 566757
[  112.266137] current process - page fault count 70
[  112.266138] dirty page of all processes:
[  112.266138] pid:1--dirty page:0
[  112.266139] pid:2--dirty page:0
[  112.266139] pid:3--dirty page:0
[  112.266140] pid:4--dirty page:0
[  112.266140] pid:5--dirty page:0
[  112.266141] pid:6--dirty page:0
[  112.266141] pid:7--dirty page:0
[  112.266142] pid:8--dirty page:0
[  112.266142] pid:9--dirty page:0
[  112.266143] pid:10--dirty page:0
[  112.266143] pid:11--dirty page:0
[  112.266144] pid:12--dirty page:0
[  112.266144] pid:13--dirty page:0
[  112.266145] pid:14--dirty page:0
[  112.266145] pid:15--dirty page:0
[  112.266146] pid:16--dirty page:0
[  112.266146] pid:17--dirty page:0
[  112.266147] pid:18--dirty page:0
[  112.266147] pid:19--dirty page:0
```

```
[  112.266136] system process - page fault count 566757
[  112.266137] current process - page fault count 70
[  112.266138] dirty page of all processes:
[  112.266138] pid:1--dirty page:0
[  112.266139] pid:2--dirty page:0
[  112.266139] pid:3--dirty page:0
[  112.266140] pid:4--dirty page:0
[  112.266140] pid:5--dirty page:0
[  112.266141] pid:6--dirty page:0
[  112.266141] pid:7--dirty page:0
[  112.266142] pid:8--dirty page:0
[  112.266142] pid:9--dirty page:0
[  112.266143] pid:10--dirty page:0
[  112.266143] pid:11--dirty page:0
[  112.266144] pid:12--dirty page:0
[  112.266144] pid:13--dirty page:0
[  112.266145] pid:14--dirty page:0
[  112.266145] pid:15--dirty page:0
[  112.266146] pid:16--dirty page:0
[  112.266146] pid:17--dirty page:0
[  112.266147] pid:18--dirty page:0
[  112.266147] pid:19--dirty page:0
```

三、讨论、心得（20 分）

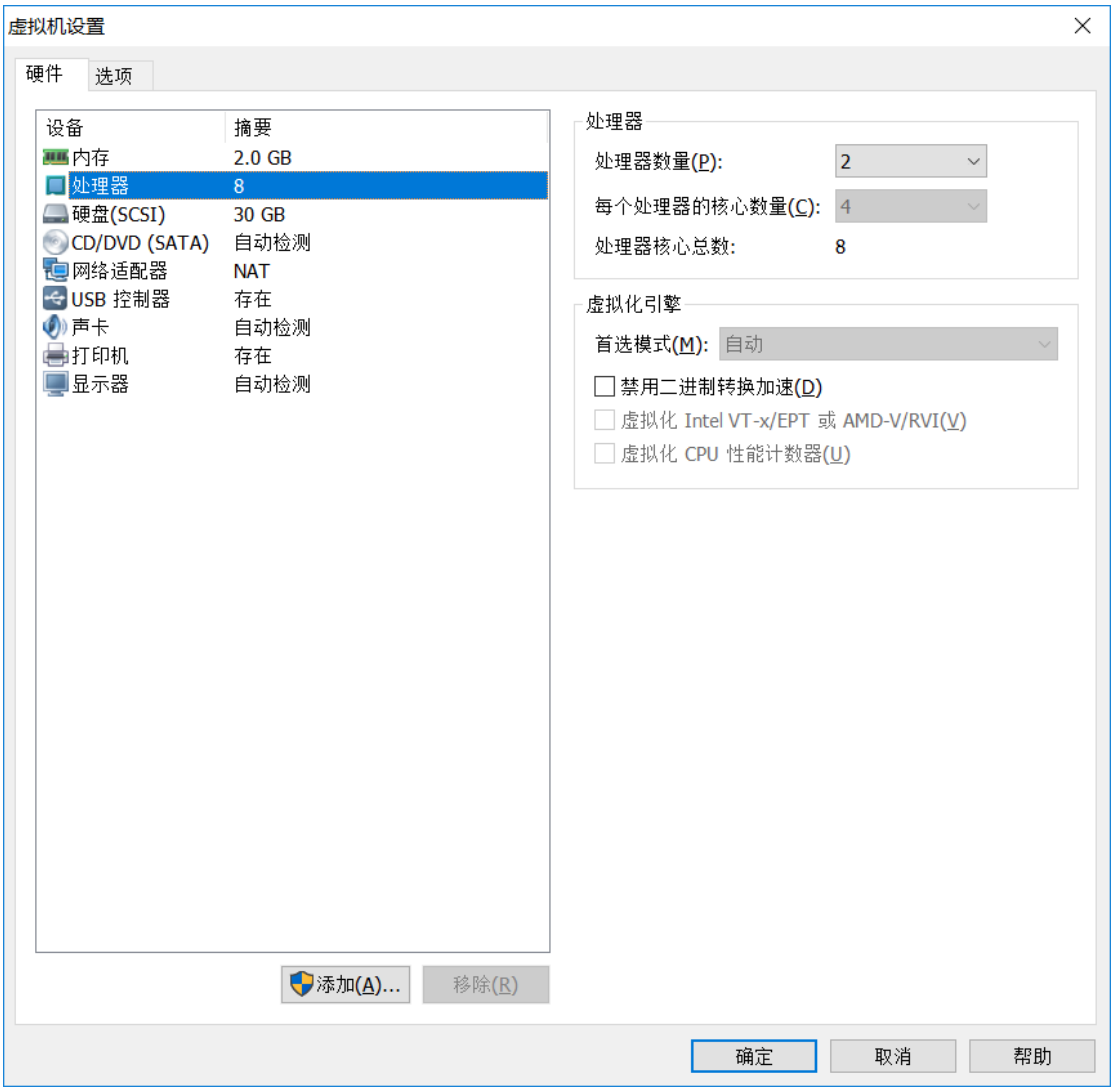
1. 一开始没有下载库，果然出现了问题：

```
zjy@ubuntu:~/build_kernel/linux-source-4.15.0$ make menuconfig
  HOSTCC  scripts/basic/fixdep
 *** Unable to find the ncurses libraries or the
 *** required header files.
 *** 'make menuconfig' requires the ncurses libraries.
 ***
 *** Install ncurses (ncurses-devel) and try again.
 ***
scripts/kconfig/Makefile:202: recipe for target 'scripts/kconfig/dochecklxdialog' failed
make[1]: *** [scripts/kconfig/dochecklxdialog] Error 1
Makefile:547: recipe for target 'menuconfig' failed
make: *** [menuconfig] Error 2
```

根据实验指示下载了 ncurses-devel 后可以正常编译。

2. 一开始虚拟机的配置是 1 核 1 线程，编译地极慢，请教同学后发现可以根据自身电脑配置更改虚拟机配置，加快编译速度，于是就更改为 2 核 4 线程，以 j8 来编

译。



3. 第一次完成二次编译内核后，在命令行输入 sudo reboot 重启，重启后虚拟机就提示禁用 CPU 无法开机，找不到原因，只好重新做了一遍实验。使用 VMware 重启，一切正常。

4. 发现问题重要统计系统缺页数，因此 sys.c 应该把 return 0 改为 return pfcount.

   原先是：

sys.c (~/build_kernel/linux-source-4.15.0/kernel) - gedit

Open Save

```
 *
 * Note, despite the name, this returns the tgid not the pid.  The tgid and
 * the pid are identical unless CLONE_THREAD was specified on clone() in
 * which case the tgid is the same in all threads of the same group.
 *
 * This is SMP safe as current->tgid does not change.
 */
SYSCALL_DEFINE0(getpid)
{
        return task_tgid_vnr(current);
}

/* Thread ID - the internal kernel "pid" */
SYSCALL_DEFINE0(gettid)
{
        return task_pid_vnr(current);
}

SYSCALL_DEFINE0(mysyscall)
{
        printk("current process - page fault count %ld \n",current->pf);
        return 0;
}


/*
 * Accessing ->real_parent is not SMP-safe, it could
 * change from under us. However, we can use a stale
 * value of ->real_parent under rcu_read_lock(), see
 * release_task()->call_rcu(delayed_put_task_struct).
 */
SYSCALL_DEFINE0(getppid)
{
        int pid;

        rcu_read_lock();
        pid = task_tgid_vnr(rcu_dereference(current->real_parent));
```

C Tab Width: 8 Ln 855, Col 2 INS

应该是：

sys.c (~/build_kernel/linux-source-4.15.0/kernel) - gedit

Open Save

```c
 *
 * This is SMP safe as current->tgid does not change.
 */
SYSCALL_DEFINE0(getpid)
{
        return task_tgid_vnr(current);
}

/* Thread ID - the internal kernel "pid" */
SYSCALL_DEFINE0(gettid)
{
        return task_pid_vnr(current);
}

SYSCALL_DEFINE0(mysyscall)
{
        printk("current process - page fault count %ld \n", current->pf);
        return pfcount;
}

/*
 * Accessing ->real_parent is not SMP-safe, it could
 * change from under us. However, we can use a stale
 * value of ->real_parent under rcu_read_lock(), see
 * release_task()->call_rcu(delayed_put_task_struct).
 */
SYSCALL_DEFINE0(getppid)
{
        int pid;

        rcu_read_lock();
        pid = task_tgid_vnr(rcu_dereference(current->real_parent));
        rcu_read_unlock();

        return pid;
}
```

C Tab Width: 8 Ln 860, Col 16 INS

而且应该修改一下用户态程序。

发现少看了一个要求，实际上应该是：

build_kernel/linux-source-4.15.0/kernel) - gedit

Open

```
 *
 * This is SMP safe as current->tgid does not change.
 */
SYSCALL_DEFINE0(getpid)
{
        return task_tgid_vnr(current);
}

/* Thread ID - the internal kernel "pid" */
SYSCALL_DEFINE0(gettid)
{
        return task_pid_vnr(current);
}

SYSCALL_DEFINE0(mysyscall)
{
        printk("system process - page fault count %ld \n", pfcount);
        printk("current process - page fault count %ld \n", current->pf);
        printk("dirty page of all processes:\n");
        struct task_struct *p = NULL;
        for(p = &init_task; (p = next_task(p)) != &init_task; )
        {
                printk("pid:%ld--dirty page:%d\n",p->pid,p->nr_dirtied);
        }
        return pfcount;
}

/*
 * Accessing ->real_parent is not SMP-safe, it could
 * change from under us. However, we can use a stale
 * value of ->real_parent under rcu_read_lock(), see
 * release_task()->call_rcu(delayed_put_task_struct).
 */
SYSCALL_DEFINE0(getppid)
{
        int pid;

        rcu_read_lock();
        pid = task_tgid_vnr(rcu_dereference(current->real_parent));
        rcu_read_unlock();

        return pid;
}

SYSCALL_DEFINE0(getuid)
{
        /* Only we change this so SMP safe */
```

5. 因为中间做错了几次，就重新编译了好几次，发现后面编译的很快。

四、问题

1. 多次运行 test 程序，每次运行 test 后记录下系统缺页次数和当前进程缺页次数，给出这些数据。test 程序打印的缺页次数是否就是操作系统原理上的缺页次数？有什么区别？

在仅打印系统缺页数和当前进程缺页数上查看多次运行的结果：

第一次运行：

```
zjy@ubuntu:~/build_kernel$ ./test
system page fault count(pfcount) = 539764
```

```
[  519.376488] current process - page fault count 67
```

第二次运行：

```
zjy@ubuntu:~/build_kernel$ ./test
system page fault count(pfcount) = 583003
```

```
[  519.376488] current process - page fault count 67
[  558.158104] current process - page fault count 69
```

第三次运行：

```
zjy@ubuntu:~/build_kernel$ ./test
system page fault count(pfcount) = 584373
```

```
[  519.376488] current process - page fault count 67
[  558.158104] current process - page fault count 69
[  726.722952] current process - page fault count 67
```

第四次运行：

```
zjy@ubuntu:~/build_kernel$ ./test
system page fault count(pfcount) = 586191
```

```
[  519.376488] current process - page fault count 67
[  558.158104] current process - page fault count 69
[  726.722952] current process - page fault count 67
[  836.631302] current process - page fault count 68
```

第五次运行：

```
zjy@ubuntu:~/build_kernel$ ./test
system page fault count(pfcount) = 586634
```

```
[  519.376488] current process - page fault count 67
[  558.158104] current process - page fault count 69
[  726.722952] current process - page fault count 67
[  836.631302] current process - page fault count 68
[  856.714645] current process - page fault count 67
```

运行次数	系统缺页数	当前进程缺页数
1	539764	67
2	583003	69
3	584373	67
4	586191	68
5	586634	67

可以看出每次运行，系统缺页次数都会明显增长，而当前进程的缺页次数则基本不变。

2. 除了通过修改内核来添加一个系统调用外，还有其他的添加或修改一个系统调用的方法吗？如果有，请论述。

除了内核编译法来添加系统调用之外，也可以通过 module 进行内核添加来添加系统调用。这种方法是 采用系统调用拦截的一种方式，改变某一个系统调用号对应的服务程序，变为自己编写的程序，从而相 当于添加了系统调用。

3. 对于一个操作系统而言，你认为修改系统调用的方法安全吗？请发表你的观点。

我认为对于一个操作系统而言，修改系统调用并不够安全。因为当修改系统调用时，有可能会对原来系 统调用表中的其他调用进行修改，从而使得其名称发生变化或者是缺少对应编号的系统调用，可能会给 调用正常系统调用接口的准确性和安全性造成威胁。