《操作系统实验》Lab3实验报告

09300240004 计算机科学与技术朱恬骅

#### 实验目的

* 熟悉Linux操作系统；
* 学习有关系统调用的知识。

#### 实验要求

* 修改Linux 0.11内核代码并重新编译；
* 在修改的内核中添加一个新的系统调用，在屏幕上输出“Hello, world!”字样。

#### 实验内容

1. 在Ubuntu 11.04系统上安装bochs：

$ sudo apt-get install bochsbochs-x

1. 解压下载的linux-0.11-devel-060625.tar.gz，修改Bochs配置文件：

$ tar -xzf linux-0.11\*.tar.gz。

Bochs配置文件内容：（加粗的为修改过的行）

|  |
| --- |
| **romimage: file=$BXSHARE/BIOS-bochs-latest**  megs: 16  **vgaromimage: file=$BXSHARE/VGABIOS-lgpl-latest**  **floppya: 1\_44="bootimage-0.11-hd", status=inserted**  floppyb: 1\_44=diskb.img, status=inserted  ata0-master: type=disk, path="hdc-0.11-new.img", mode=flat, cylinders=410, heads=16, spt=38  **boot: a**  log: bochsout.txt  vga\_update\_interval: 300000  keyboard\_serial\_delay: 200  keyboard\_paste\_delay: 100000  mouse: enabled=0  private\_colormap: enabled=0  fullscreen: enabled=0  screenmode: name="sample"  i440fxsupport: enabled=0 |

1. 启动Bochs，进入Linux 0.11系统，屏幕输出：

|  |
| --- |
| Please visit :  . http://bochs.sourceforge.net  . http://www.nongnu.org/vgabios  Bochs VBE Display Adapter enabled  Bochs BIOS - build: 06/30/10  $Revision: 1.247 $ $Date: 2010/04/04 19:33:50 $  Options: apmbiospcibiospnpbioseltorito rombios32  ata0 master: Generic 1234 ATA-6 Hard-Disk ( 121 MBytes)  Press F12 for boot menu.  Booting from Floppy...  Loading system...  Partition table ok.  39189/62000 free blocks  19522/20666 free inodes  3462 buffers = 3545088 bytes buffer space  Free mem: 12582912 bytes  Ok.  [/usr/root]# |

1. 进入 /usr/src/linux/ 目录，使用vi修改Linux 0.11源代码，以添加新的系统调用mycall：
   1. 在include/unistd.h中添加#define \_\_NR\_mycall 87，并在下面声明函数原型：

|  |
| --- |
| int mycall(); |

* 1. 在include/linux/sys.h中添加:extern int mycall();然后在sys\_call\_table中最后加上mycall；
  2. 在kernel/sys.c中添加函数实现如下：

|  |
| --- |
| int mycall(){  printk("Hello, world!");  return 123;  } |

* 1. 在kernel/system\_call.s中对系统调用号加1；

1. 编译内核make clean; make all; make Image; make disk
2. 重启系统，编写hello.c：

|  |
| --- |
| #include <stdio.h>  #define \_\_LIBRARY\_\_  #include <unistd.h>  \_syscall0(int, mycall);  int main()  {  int x;  x = mycall();  printf("%d", x);  return 0;  } |

1. 用修改过的include/unistd.h覆盖掉/usr/include下的unistd.h：

# cp include/unistd.h /usr/include/unistd.h

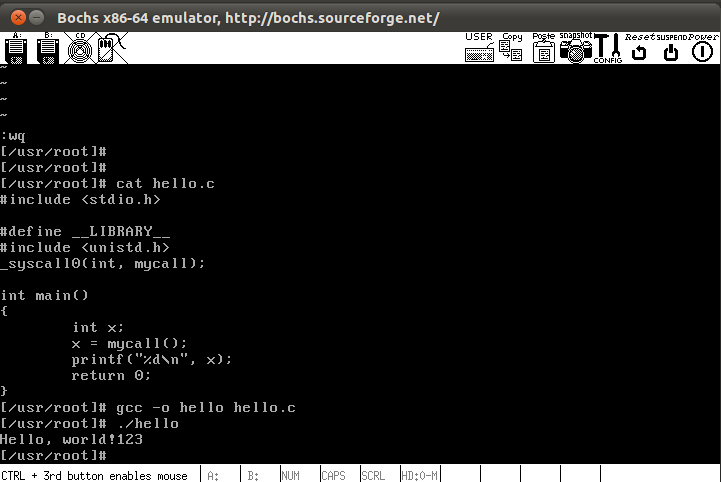
1. 编译，运行：

# gcc -o hello hello.c

# ./hello

屏幕输出：

Hello, world!123



这就完成了实验。

#### 实验中遇到的问题和解决方法

1. 重新编译好的内核无法确认是否加载。
   1. 运行./hello，输出-1，经查阅资料[[1]](#footnote-2)，猜测可能是新的系统内核未加载，于是搜索资料，无果。查看Makefile，发现有一个disk参数，尝试，无效果。
   2. 修改init/main.c，在printf(“Free Mem…”)语句后增加一行“printf(“mycall added.”)”，重新编译，重启，无显示，由此确认新内核并未加载。
   3. 重新查看Makefile，发现dd的参数中和网上查到的不一样，将“PS0”改为“fd0”后，重新运行make disk，重启，显示“mycall added.”。
2. 网上资料显示，hello.c中要定义\_\_NR\_LIBRARY\_\_，但是在我的系统中，这样定义会导致编译出错，提示问题出在\_syscall0。查看include/unistd.h，发现要使用\_\_LIBRARY\_\_。
3. 要用修改过的include/unistd.h覆盖掉/usr/include下的unistd.h，否则hello.c编译不通过。
4. 在kernel/system\_call.s中对系统调用号加1后，运行./hello仍然输出-1，这时只好再加1，重新编译运行，正常。可能是原先的代码之间存在不一致。

#### 实验收获

1. 学习了如何在Ubuntu下安装、运行Bochs，以及配置从Bochs启动Linux 0.11；
2. 学习了vi的使用；
3. 学习了添加系统调用和如何使用系统调用；
4. 初步了解了Makefile。

2011年9月24日

1. http://topic.csdn.net/u/20090831/10/4a0e5694-7873-4737-916b-02b1a7bf0f0a.html [↑](#footnote-ref-2)