针对Linux x86_64内核，如何自己写系统调用

写一个helloworld系统调用

写Linux系统调用是Linux编程中经常的需求，本文将详细讲述这个过程，需要说明的是：**本文针对Linux Ubuntu 64位操作系统，32位不支持**

1. 依赖安装

```
sudo apt-get update  #更新
sudo apt-get upgrade
sudo -s
apt-get install gcc
apt-get install python-pip python-dev libffi-dev libssl-dev libxml2-dev libxslt1-dev libjpeg8-dev zlib1g-dev
apt-get install libncursesw5-dev
```

2. 下载一个内核版本

在https://www.kernel.org/ 下载一个需要的Linux内核版本（究竟选用那个Linux系统版本，取决于你自己的需求），并解压该文件。

3. 写helloworld系统调用

```
cd ***.x   #进入到Linux内核解压后的根目录下
mkdir helloworld
cd helloworld
```

```
vim helloworld.c
```

在helloworld.c中写入下面的内容：

```
#include <linux/kernel.h>
asmlinkage long sys_helloworld(void){
    printk("Hello World~\n");
    return 0;
}
```

接着执行

```
vim Makefile  #和helloworld.c在同一个目录下
```

在Makefile文件中写入下面的内容：

```
obj-y  := helloworld.o
```

接着回到Linux内核源码的根目录中，修改Linux内核自带的Makefile文件，在其中写入下面内容：

```
core-y += kernel/.../ helloworld/   #只有helloworld是需要我们添加的，其它的在Linux本身的Makefile文件中就全部都有
```

接着执行

```
cd include/linux
vim syscalls.h
```

在打开的文件的最后面写入下面内容

```
```

```
asmlinkage long sys_helloworld(void);
```

接着回到Linux内核根目录，并执行下面指令

```
cd arch/x86/entry/syscalls
vim syscall_64.tbl
```

在打开的文件中写入下面内容：

```
333  64 helloworld  sys_helloworld   #333 is index of our system call
```

回到Linux内核的根目录

```
make menuconfig   #保存即可
make oldconfig
make -j 4  #编译内核
make modules_install install #安装新的内核
reboot   #重启计算机
uname -a   #查看是否进入到新内核中
```

4. 写C语言程序查看成功插入helloworld系统调用模块

```
//将该文件命名为test.c，并写入下面的代码，测试helloworld系统调用是否能用
#include <stdio.h>
#include <linux/kernel.h>
#include <sys/syscall.h>
#include <unistd.h>

int main(){
    long int s = syscall(333);   //333 is index of helloworld system call
    printf("System call : sys_helloworld : return %1d\n" , s);
    return 0;
}
```

写完test.c之后编译并运行，查看结果，如果return后面输出的值为0，说明上述系统调用完全正确。

```
gcc test.c
./a.out  // 输出return 0 ，值为0说明所有的系统调用都是成功的
dmesg   #查看kernel日志，最后一行看到存在helloworld，说明成功写出helloworld系统调用
```

参考资料

1. 最详细的方法 https://www.tldp.org/LDP/lkmpg/2.6/html/lkmpg.html
2. https://linux.cn/article-9628-1.html
3. http://www.cnblogs.com/hazir/p/three_methods_of_syscall.html
4. http://mooc.study.163.com/course/1000072000#/info
5. https://www.linux.it/~rubini/docs/ksys/