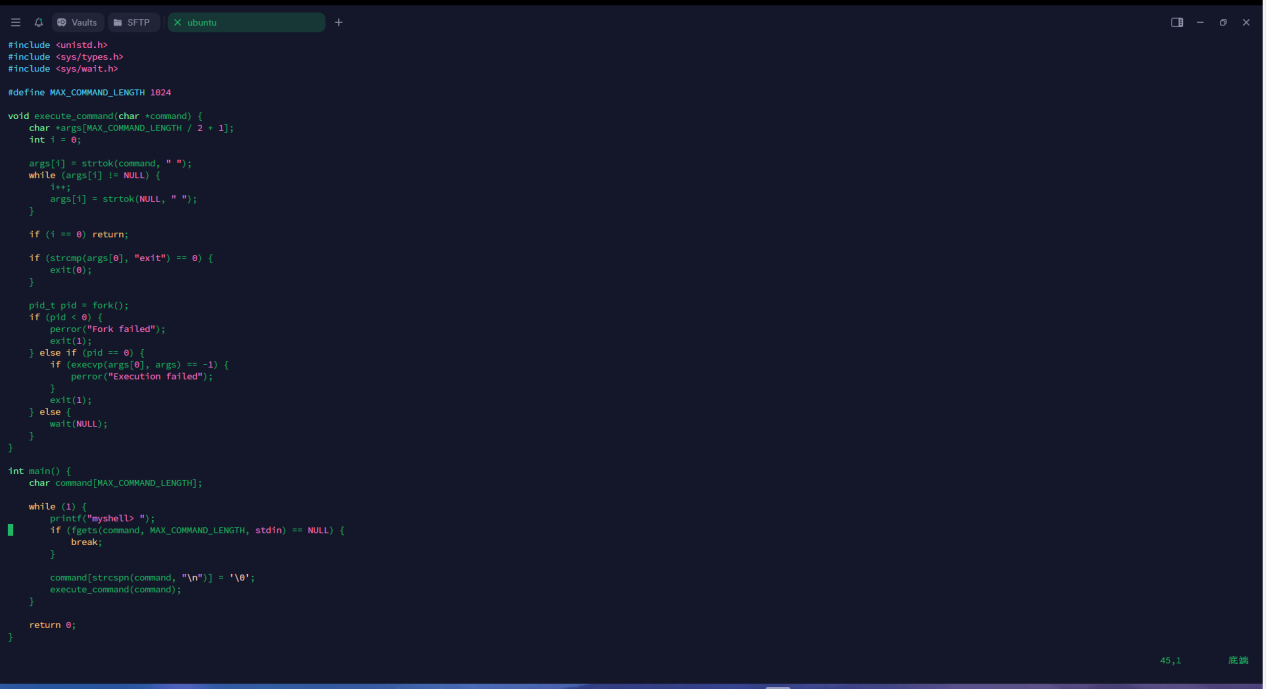
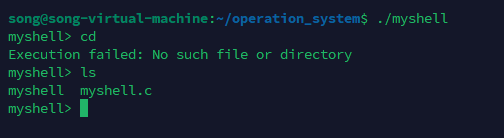
操作系统Project

1. 实现一个自己的shell

本project基于Ubuntu18.04实现，首先在终端中创建对应文件夹，编写代码，实现一个简单的shell，能够接受用户输入并执行基础命令（如ls、pwd等）

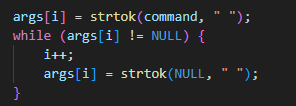




读取用户命令：

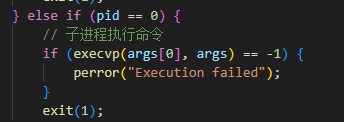
fgets(command, MAX\_COMMAND\_LENGTH, stdin)：从标准输入中读取用户输入的命令，并存储在 command 字符数组中。

解析用户输入的命令：



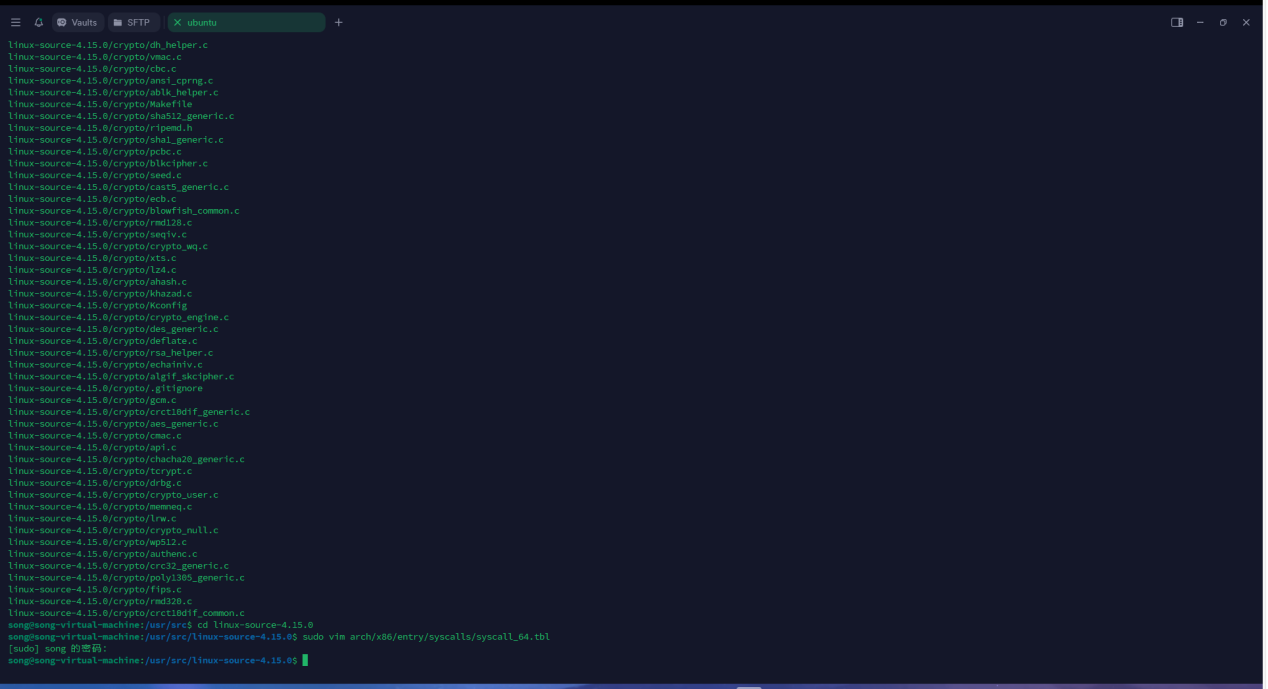
Strtok以空格为分隔符，将用户输入拆分为多个部分。第一个部分（args[0]）是命令名，比如ls；后续部分（args[1]、args[2]等）是该命令的参数，解析后的 args 是一个以 NULL 结尾的字符串数组，可以直接传递给 execvp 执行命令。

对于普通命令，代码首先通过fork创建子进程，子进程通过execvp执行用户输入的命令

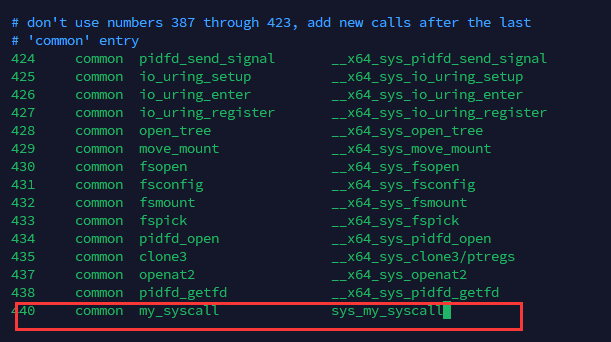


1. 向Linux系统中添加一个系统调用，该系统调用向用户空间返回字符串”hello world!”

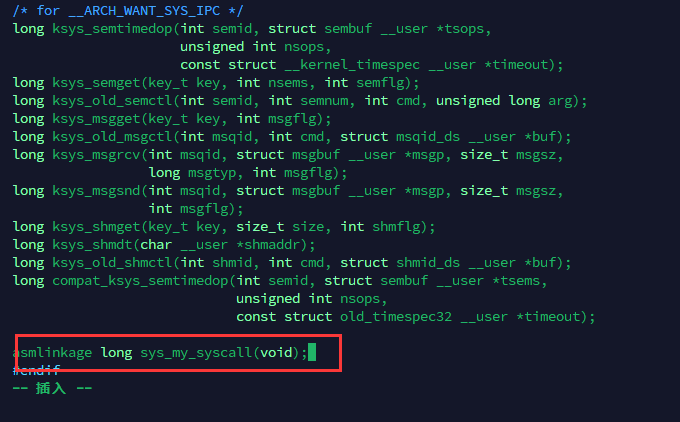
首先在Ubuntu中安装Linux内核源代码：



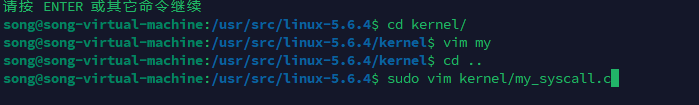
修改系统调用表，在内核源代码目录中打开arch/x86/entry/syscalls/syscall\_64.tbl文件，在文件中添加新系统调用：



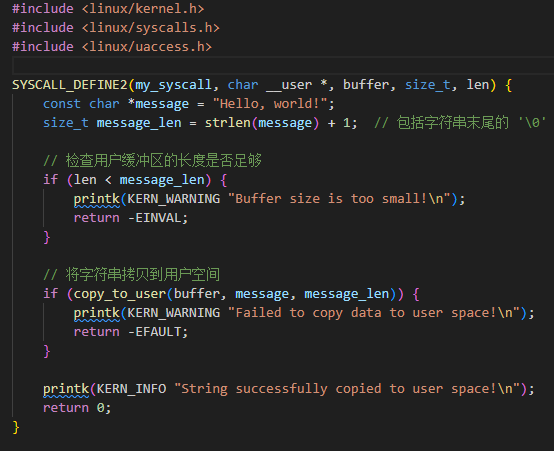
在syscalls.h中声明系统调用



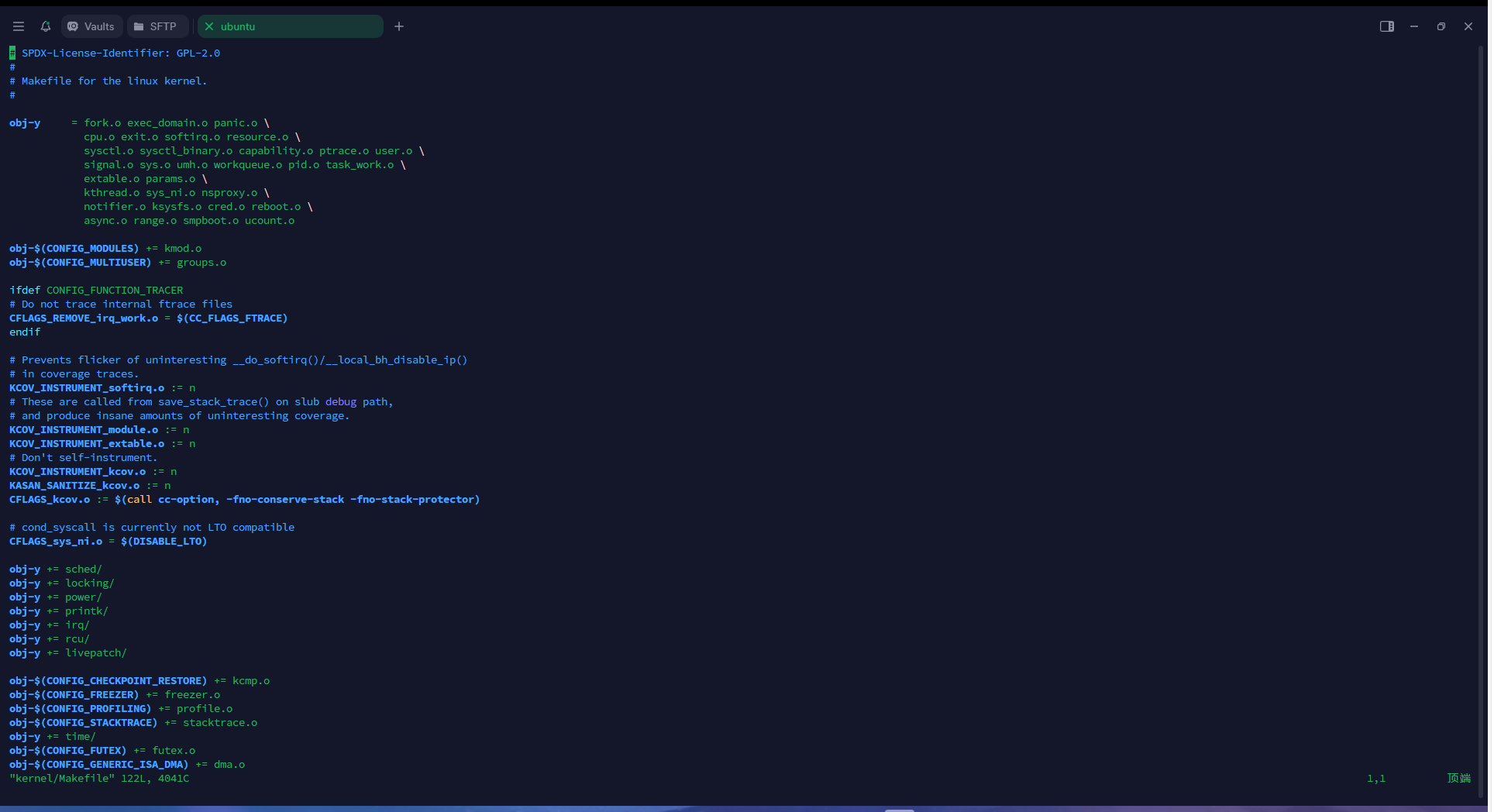
接下来添加系统调用的实现，在kernel/目录下创建一个新的文件：my\_syscall.c：



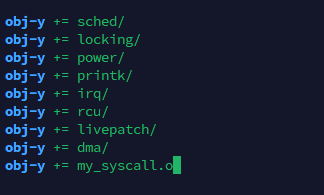
在文件中添加系统调用的代码：



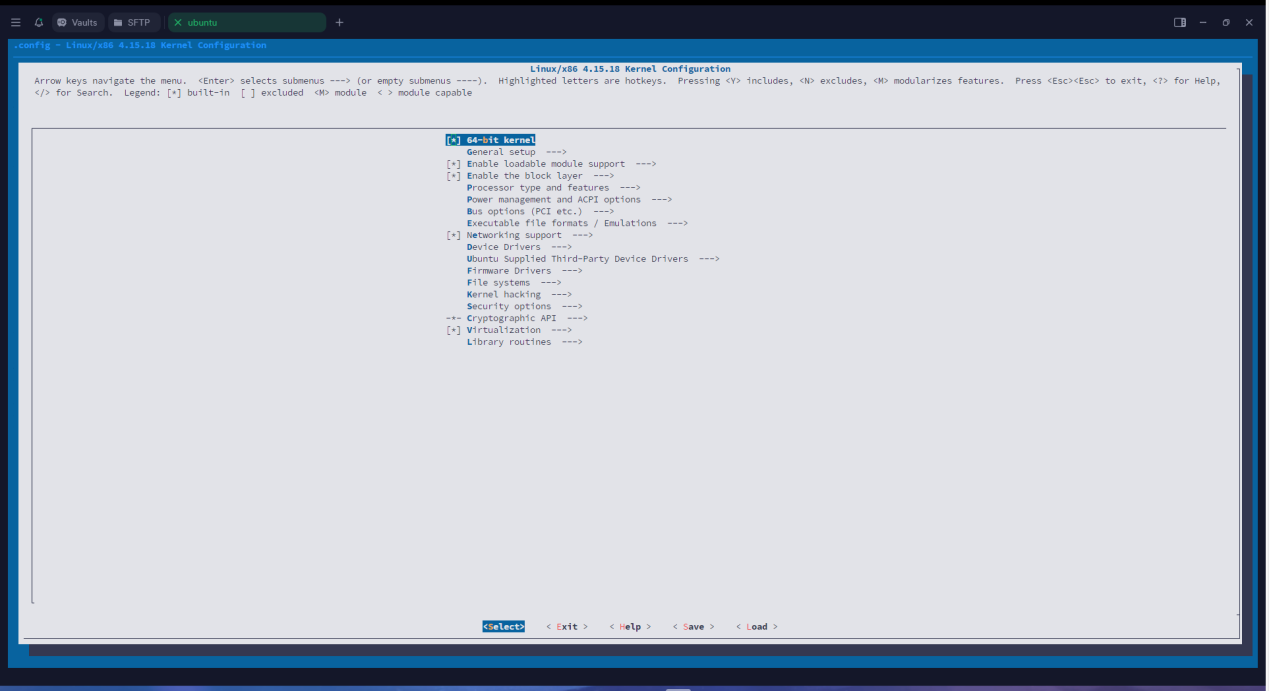
接下来修改Makefile，编译新的系统调用，打开kernel/Makefile文件：



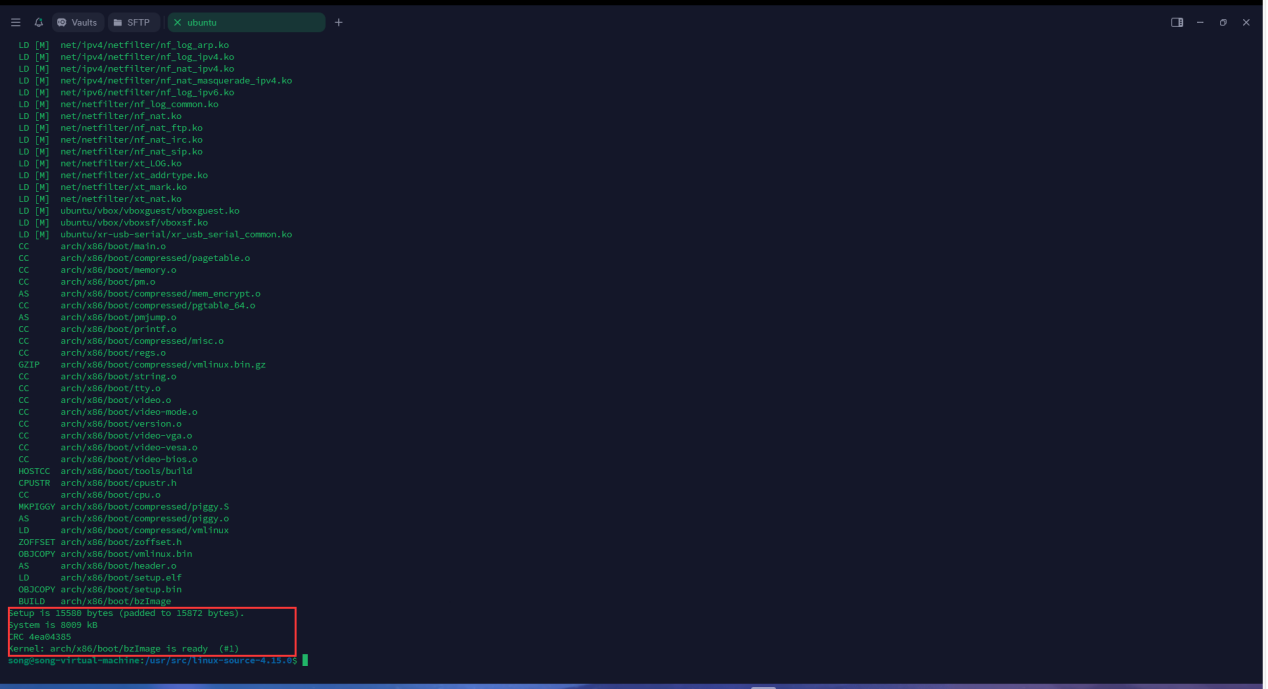
添加新增的my\_syscall.o文件到编译列表



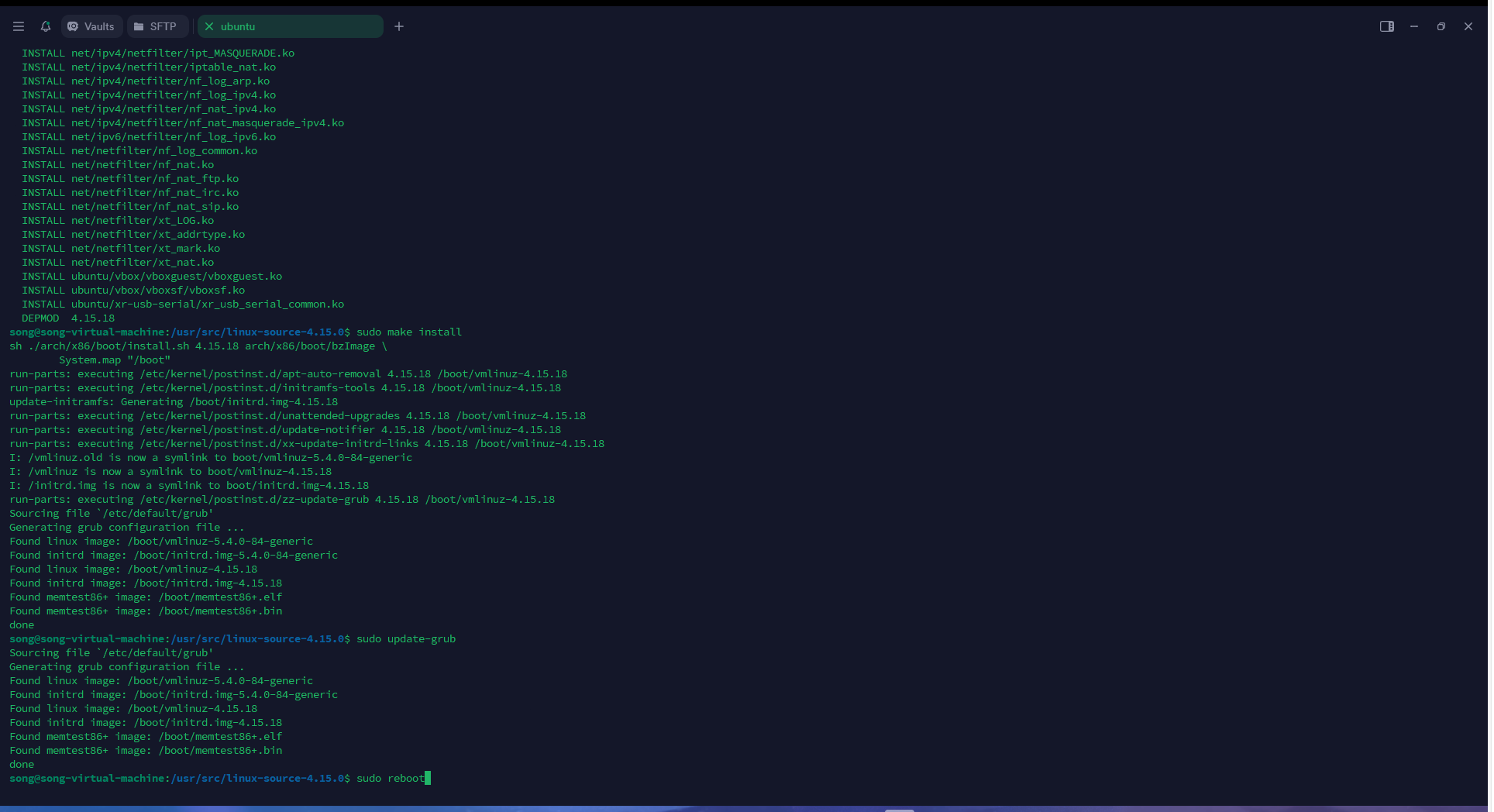
最后进行内核的编译和安装，使用menuconfig配置内核：sudo make menuconfig



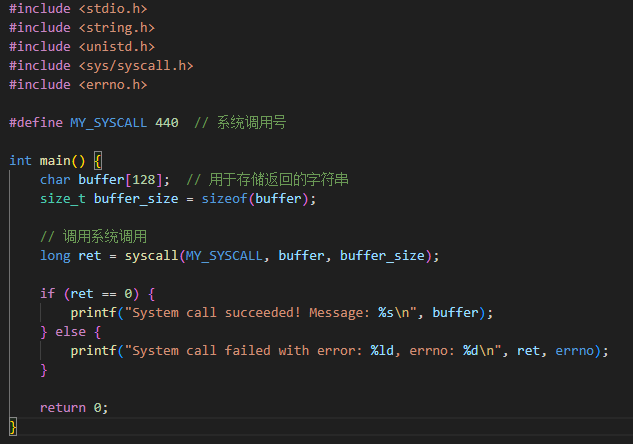
使用make -j$(nproc)编译内核，经过一段时间的等待，内核编译完成



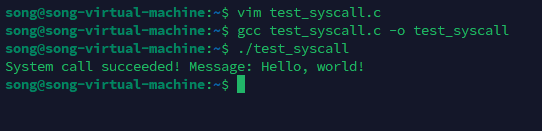
安装编译好的内核，更新GRUB并重启



下面对新增的系统调用进行测试，首先编写测试代码



编译运行结果如下：



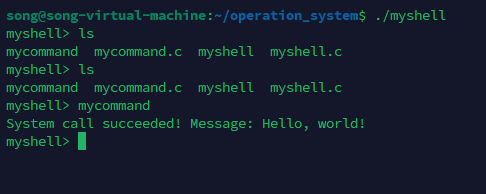
输出结果符合预期。

1. 编写一个自定义用户命令“mycommand”。在第一步中实现的shell中执行该命令。mycommand命令将通过调用第二步中定义的系统调用获得“hello wold！”字符串，并在标准输出设备中显示

只需要在原来的myshell代码中添加mycommand函数：



运行myshell的可执行程序结果如下：



输出内容符合预期，实现了用户命令进行系统调用