

PA0 实验报告

匡亚明学院：刘志刚

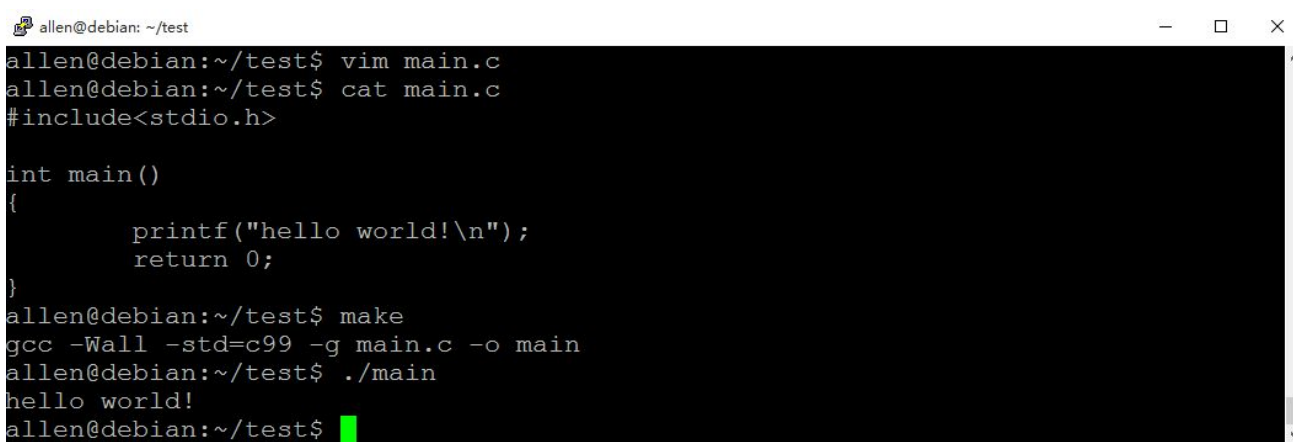
学号：141242022

实验进度:

该干的都干了，git 还在学习中。

若干截图

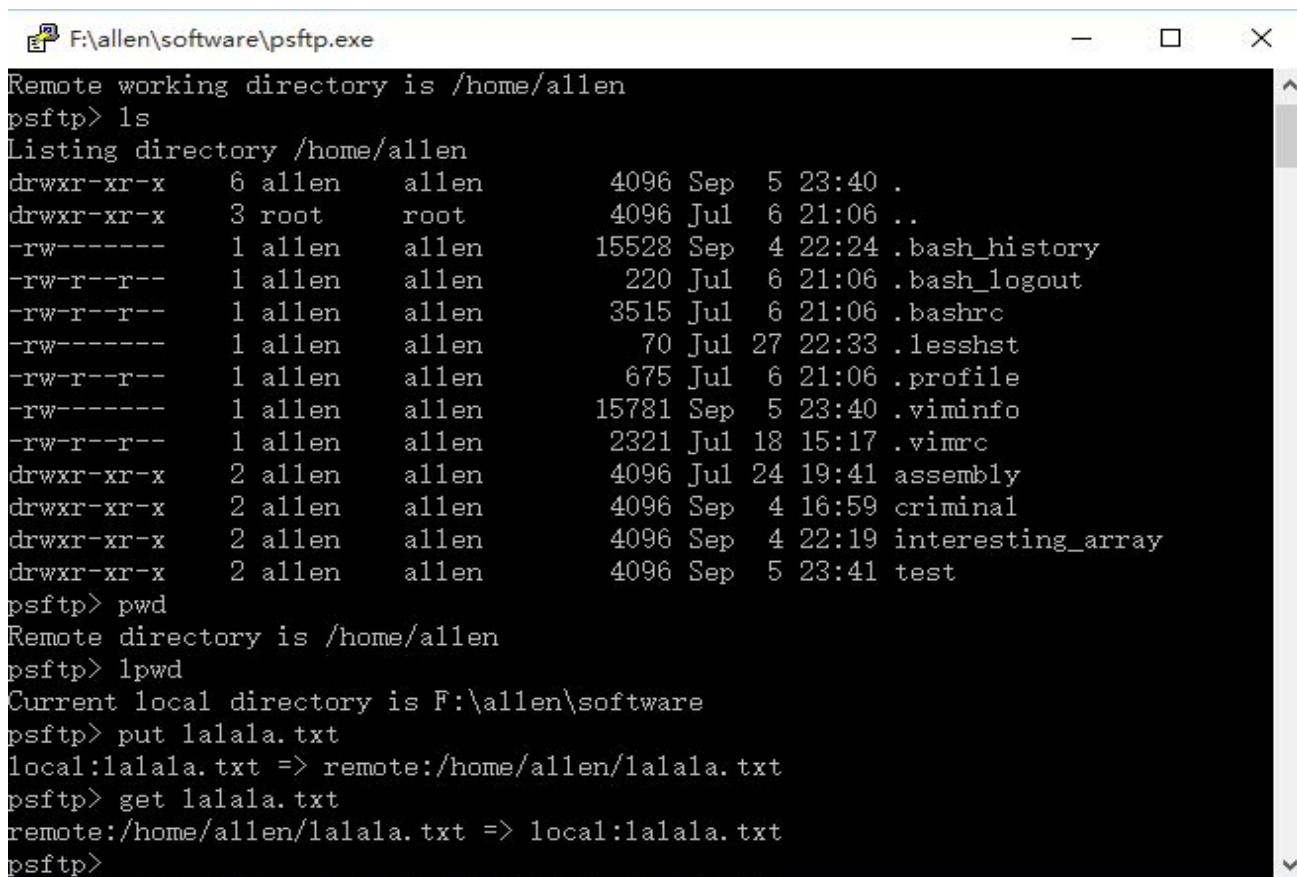
Hello world!



```
allen@debian: ~/test
allen@debian:~/test$ vim main.c
allen@debian:~/test$ cat main.c
#include<stdio.h>

int main()
{
    printf("hello world!\n");
    return 0;
}
allen@debian:~/test$ make
gcc -Wall -std=c99 -g main.c -o main
allen@debian:~/test$ ./main
hello world!
allen@debian:~/test$
```

使用 psftp 传送文件



```
F:\allen\software\psftp.exe
Remote working directory is /home/allen
psftp> ls
Listing directory /home/allen
drwxr-xr-x  6 allen  allen    4096 Sep  5 23:40 .
drwxr-xr-x  3 root   root     4096 Jul  6 21:06 ..
-rw-r--r--  1 allen  allen   15528 Sep  4 22:24 .bash_history
-rw-r--r--  1 allen  allen    220 Jul  6 21:06 .bash_logout
-rw-r--r--  1 allen  allen   3515 Jul  6 21:06 .bashrc
-rw-r--r--  1 allen  allen     70 Jul 27 22:33 .lessht
-rw-r--r--  1 allen  allen    675 Jul  6 21:06 .profile
-rw-r--r--  1 allen  allen   15781 Sep  5 23:40 .viminfo
-rw-r--r--  1 allen  allen    2321 Jul 18 15:17 .vimrc
drwxr-xr-x  2 allen  allen    4096 Jul 24 19:41 assembly
drwxr-xr-x  2 allen  allen    4096 Sep  4 16:59 criminal
drwxr-xr-x  2 allen  allen    4096 Sep  4 22:19 interesting_array
drwxr-xr-x  2 allen  allen    4096 Sep  5 23:41 test
psftp> pwd
Remote directory is /home/allen
psftp> lpwd
Current local directory is F:\allen\software
psftp> put lalala.txt
local:lalala.txt => remote:/home/allen/lalala.txt
psftp> get lalala.txt
remote:/home/allen/lalala.txt => local:lalala.txt
psftp>
```

我自己的简单的 vim 配置以及 tmux 及 gdb 使用截图

```
54 set nu
55 set shiftwidth=4
56 set tabstop=4
57 set softtabstop=4
58 filetype on
59 autocmd FileType awk,sh :set ai
60 autocmd FileType c,cpp :set cindent
61 autocmd FileType py :set smartindent
```

61,1

Bot

```
1 #include<vector>
2 #include<algorithm>
3 #include<cstdio>
4 #include<climits>
5
6 using std::vector;
7 using std::sort;
8
9 class Enemy_pair
10 {
11
12     public:
13         Enemy_pair(int p1,int p2,int hatred_value)
14         {
15             a=p1;
16             b=p2;
```

"main.cpp" 101L, 2045C

1,1

Top

<<http://www.gnu.org/software/gdb/bugs/>>.

Find the GDB manual and other documentation resources online at:

<<http://www.gnu.org/software/gdb/documentation/>>.

---Type <return> to continue, or q <return> to quit---

For help, type "help".

Type "apropos word" to search for commands related to "word"

...

Reading symbols from main...done.

(gdb) break main

Breakpoint 1 at 0x8048e22: file main.cpp, line 88.

(gdb) run < input

Starting program: /home/allen/criminal/main < input

Breakpoint 1, main () at main.cpp:88

88 scanf("%d%d",&n,&m);

(gdb)

个人关于 GUI 与 CLI 的一些想法

GUI 和 CLI 各有何优劣？

图形界面

优势：

- 图形界面对用户很友好，降低了用户的学习成本。视觉信息让人容易接受，同时降低了误操作的可能性。我认为图形界面的是计算机发展史上极其伟大的发明（也难怪苹果的 Macintosh 会那样受欢迎），它使得计算机真正成为人人可用的，帮助人类工作的机器。

劣势：

- ◆ 在应对复杂操作时力不从心（很难想象如何在一个屏幕上安排下几十个选项）
- ◆ 基本以二进制数据流为主，程序相互之间难以交互。所以程序基本上都是大而全的节奏。
- ◆ 要花一定量的时间熟悉菜单功能布局，而且不同程序甚至不同相同程序不同版本的布局可能不同，迁移有可能有些困难。比如从 VC++ 到 VC2008 到 VS2013，热键变了，选项变了，我按了半天 F7，它就是不编译，我也是醉了。

命令行界面

优势：

- 你假如熟练了，可以极大地提高效率。（前提是你熟练了）
- 在兼容 POSIX 系统下，工具集都基本通用，这就意味着你除了不能在 Windows 上干之外，你在使用其他所有现代操作系统时，学习成本几乎为 0。
- 管道的使用，以及无处不在的 Unix 哲学使得组织这些程序很容易。

命令行的劣势：

- ◆ 上手慢，反人类。
- ◆ 每一步都要动脑筋理解，而且这些命令干完活之后默认啥都不说，所以一不小心做错了什么，半天才反应过来，然而此时想剁手已经来不及了。

为何 Windows 比较 fat 而 Linux 可以很 slim?

不知此 fat 是指什么 fat? 似乎 Linux 内核胖多了（虽然里面大部分都是驱动程序）。Windows 相对较胖，估计还是要支持的服务比较多，而且 Windows 的图形界面与系统耦合较为紧密，很难裁剪，所以相对较大吧。细节的我也不懂，不过在某种意义上，Windows 胖也有可能是因为 Windows 兼容性这个包袱大一些，老服务要支持，不然开发者不干。新服务又要不停的开发，新的语言，新的运行环境都要支持，所以胖是必然的。感觉系统升级，内核大版本升级，甚至从 32 位到 64 位的大迁移，绝大多数的程序都可以照跑不误，也难怪 windows 会胖。

为什么像 poweroff, reboot 这样的指令需要 superuser 权限？如果普通用户也能执行，会有什么后果？

如果不用 superuser 权限，不然一个普通用户分分钟可以利用关机结束所有人的工作。而一个普通用户不能管理其他用户的计算资源，因此 poweroff 命令在设计时就应该需要 superuser 权限。

为什么我们要用 sudo 来代替 su?

由于使用 su 切换到超级用户后普通用户拥有无限制的权限，所以 su 并不适

合有多人使用及管理的系统。如果用 `su` 来切换到超级用户来管理系统，也不能明确哪些工作因由哪些人来完成。太危险了！

通过 `sudo`，我们能把某些超级权限有针对性的下放，并且不需要普通用户知道 `root` 密码，所以 `sudo` 相对于权限无限制性的 `su` 来说，还是比较安全的，所以 `sudo` 也能被称为受限制的 `su`；另外 `sudo` 是需要授权许可的，所以也被称为授权许可的 `su`。安全多了！

为什么不应该使用 `root` 用户来做实验？

能力越大，责任越大。

如果你不能一直保持清醒，拿着屠龙宝刀会一不小心伤了自己。

养成上机好习惯，要从娃娃抓起。

关于 `make`

和其他自动化的版本管理以及项目软件应该是一样的吧，除了他要求我们显示的表明每个文件之间的依赖关系。然后 `make` 根据依赖关系把更新过的源文件重新编译一遍，否则就返回 `***is up to date`。我猜 `make` 是基于时间戳来判断文件是否是最新的，因为只要把源文件 `touch` 一遍之后，`make` 就会完全重新编译一遍。

实践过程中遇到的问题，以及解决办法：

在 `select and install software` 时挂掉了，后来参考 `gitbooks` 上的说明，断网就好了。

我不知道怎样在主机与虚拟机之间传输文件，于是我参考 `gitbooks` 上面的内

容，实践了一遍又一遍，终于好了。

我不知道怎样使用 `tmux`，于是上网搜了一篇教程，于是会了。

仍然存在的问题：

装第二台虚拟机时，SSH 配置始终没成功，尤其是里面那几个 ip 地址的设置，不知道该怎么办。

在实践中我的体会：

原来不装桌面环境也能混，还可以混得很好，我以前都只敢 `graphical install`，而且都是用的 Ubuntu，臃肿得不行。现在终于体会到什么叫做 Linux 的轻快简洁了，虚拟机跑起来丝毫不卡。

我猛然回想起上学期刚入门时痛苦操作的情形，再想想现在能熟练地在虚拟机上写代码、调试代码，才发现当初的作死都是值得的。想想科大开源镜像站主页上的那句“Another Infinity”，才发现很对啊，我已初步迈入了那片新世界。

感谢 NJUOPEN 社团技术部的汪天泽部长，帮我装了那么多次电脑，带我走入了这个世界。