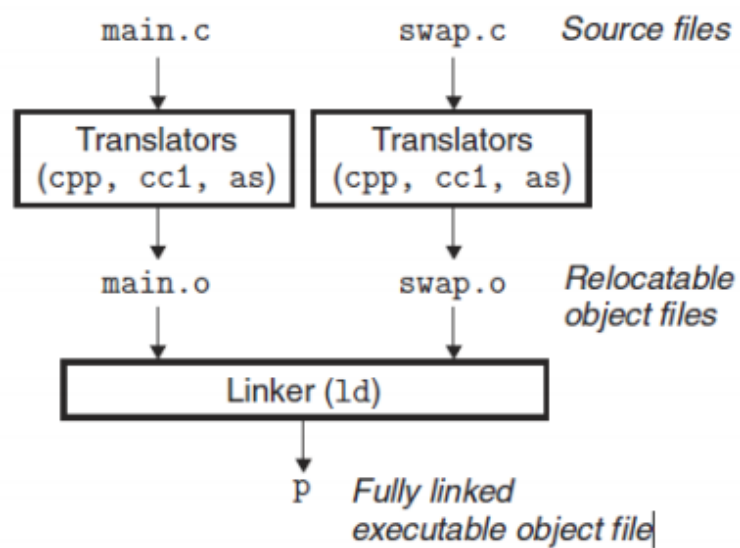


Homework 11

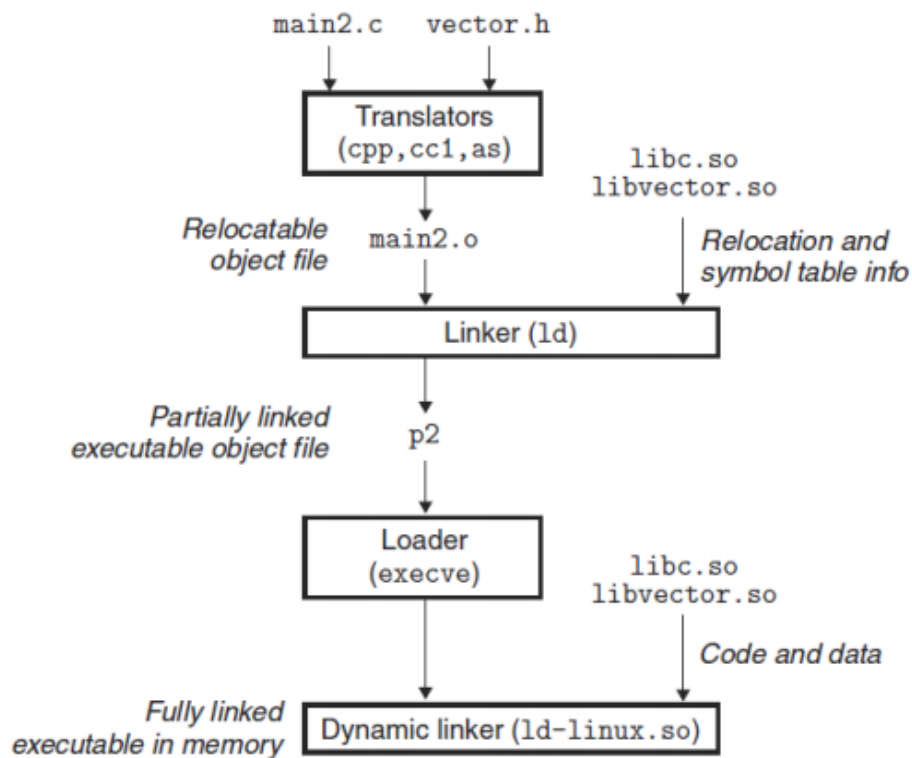
Problem 1

Express the difference between the procedure of static link and the procedure of dynamic link (with shared library). You can draw a picture to show that.

Static link:



Dynamic link:



Problem 2

- 1) Draw a Linux run-time memory image to show the location of GOT, PLT and shared library.
- 2) Express the function of PLT and GOT. And how to use them?
- 3) When the GOT is generated, and when the items of it are relocated?(Two situations: PIC Data References and PIC Function Calls)

1)

内核虚拟存储器
用户栈
共享库(shared library)
堆
读/写数据段
GOT (.data 中)
只读的代码和数据
PLT (位于.text 中)

2) GOT 作用是把位置无关的地址重定位到绝对地址; PLT 的作用是把位置独立的函数调用重定向到绝对位置。

PIC 数据引用时, 通过位置无关代码, 使程序跳转到 GOT 表中的相应条目, 通过 GOT 间接引用每个全局变量。

PIC 函数调用时, 将被调用函数绑定到 PLT 表中的相应条目中, 并从条目的第一条指令开始执行, 从而得到绝对地址。

3) GOT 表由静态链接器生成, 在动态链接时由动态链接器重定位。

Problem 3

Suppose we have **main.c** and a shared library **foo.so**. In **main.c**, we called a function "void flag()" which is in the **foo.so**.

Complete the following Codes in **main.c** to call the **flag()**.

```
#include <stdio.h>
#include <stdlib.h>
#include <dlfcn.h>

Int main()
{
    // Your codes here.
    void *handle;
    void (*flag)();
    char *error;

    handle = dlopen("./foo.so", RTLD_LAZY);
    if(!handle){
        fprintf(stderr,"%s\n",dlerror());
        exit(1);
    }

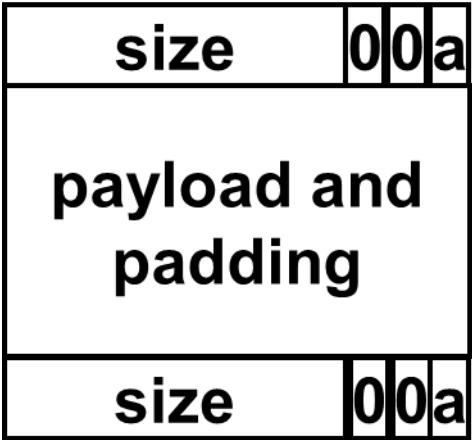
    flag = dlsym(handle,"flag");
    If((error=dlerror()) != NULL){
        fprintf(stderr,"%s\n",error);
        exit(1);
    }

    flag();

    return 0;
}
```

Problem 4

1. Boundary tags. Block structure:



2.

8/000							8/000	6/001					6/001
-------	--	--	--	--	--	--	-------	-------	--	--	--	--	-------