LinkLab 草案

ICS1-2024

HuanChengFly

2024-07-09

中国人民大学

Outline

1. 实验概述
   - 实验目标
   - 实验内容
2. 设计草案
   - 草案 1：a.out ❌
   - 草案 2：类 FLE ✓

Outline

1.1 实验目标

帮助同学们更好地理解程序链接的过程

Why?

1.1 实验目标

帮助同学们更好地理解程序链接的过程

Why?

- 链接很重要!
  - 对部署 C++ 项目、排查错误都有帮助
  - 作为一个独立的章节却没有 Lab……
- 之后的课程中不会再涉及到链接的内容
- 纯理论的讲解过于抽象
  - 大部分同学在学习后对链接的理解仍较为粗浅

1.2 实验内容

实现一个简易的链接器

主要是符号解析 & 重定位

只涉及静态链接 **动态链接过于复杂**

1.2 实验内容

实现一个简易的链接器

主要是符号解析 & 重定位

只涉及静态链接 **动态链接过于复杂**

How?

ELF 文件?

- 工业级标准，不易理解

重点：通过一个 naive 的故事让同学们理解链接的理念，在动手中一步步掌握 why 和 how 的问题

Outline

2.1 草案 1：a.out ❌

早期的 Unix 系统使用的可执行文件格式

优点：真实的格式！但 Linux 5.18 开始被正式移除支持：

- 无法直接运行 a.out 格式的可执行文件
- GCC 无法生成 a.out 格式的文件
- binutils 也不再支持 a.out 格式……

需要额外造轮子，同时并不 Human-friendly

2.2 草案 2：类 FLE ✔

Reference: 可执行文件和加载 by jyy@NJU

```
#!./exec
{
    "type": ".exe",
    "symbols": {
        "_start": 15
    },
    ".load": [
        "🔢: ff ff ff ff ff ff ff",
        "🔢: ff ff ff ff ff ff ff",
        "🔢: 48 c7 c0 3c 00 00 00",
        "🔢: 48 c7 c7 2a 00 00 00",
        "             ^",
        "             |",
        "          This byte is return code (42).",
        "🔢: 0f 05 ff ff ff ff ff",
        "🔢: ff ff ff ff ff ff ff"
    ]
}
```

Shebang

2.2 草案 2：类 FLE ✔

主要组成部分：

- 代码
- 符号
- 重定位

实现：

- 编译器（调用 gcc 并将 gcc 输出的 .o 文件转换为 FLE 格式）
- 链接器（Lab 的内容，同学们自行实现，进行符号解析和重定位）
- 加载器（`mmap` and run!）

2.2 草案 2：类 FLE ✔

优点：

- 人类友好

缺点：

- 过于“人类友好”……
  - 怎么解析和处理 JSON？（尤其是 C/C++）