第7章 链接

Compiler Driver

- Source files:
  - *.c (ASCII source file)
- Translators:
  - cpp(C preprocessor) -> *.i (ASCII intermediate file)
  - ccl(C compiler) -> *.s (ASCII assembly-language file)
  - as(assembler) -> *.o (relocatable object file)
- Linker:
  - ld(linker program) -> *.elf (executable object file)

Static Linking

- Symbol resolution
- Relocation

Object File

- 分类：
  - Relocatable object file
    - Windows -> *.obj
    - Linux -> *.o
  - Executable object file
    - Windows -> *.exe
    - Linux -> /bin/bash
  - Shared object file
    - Windows -> *.dll
    - Linux -> *.so
- Format (COFF)
  - Windows -> Portable Executable(PE)
  - MacOS-X -> Mach-O
  - x86-64 Linux & Unix -> Executable and Linkable Format(ELF)
- Relocatable Object Files
  - ELF header:
    1. 生成该文件的系统的字的大小和字节顺序
    2. ELF header的大小
    3. 目标文件的类型（relocatable， executable or shared）
    4. 机器类型（e.g. x86-64)
    5. section header table的文件偏移，以及其中entry的大小和数量
  - .text:(Ndx=1)
    - 已编译的机器代码
  - .rodata:
    - 只读数据
  - .data:(Ndx=3)

        - 已初始化的全局和静态变量
    - .bss:（Better save space）
        - 未初始化的全局和静态变量
        - 初始化为0的全局和静态变量
        - 占位符，不占据实际空间；运行时，内存分配，初始值为0
    - .symtab:
        - 符号表：定义和引用的函数和全局变量的信息
        - 类型：
            - 全局符号： 非静态函数和全局变量
            - 外部符号： 引用的非静态函数和全局变量
            - 局部符号： static属性的函数和全局变量（对外部模块不可见）
        - Entry:
            - name: String table offset
            - type: Function or Data
            - binding: Local or Global
            - value: Section offset or absolute address
            - section: <=> Ndx
                - pseudosection:（仅出现在可重定位目标文件）
                    - ABS: 不该被重定位的符号
                    - UNDEF： 未定义符号
                    - COMMON： 未被分配位置的未初始化数据（未初始化的全局变量，其他是.bss），也即弱全局符号
        - Resolution:
            - Rules:（Linux链接器）
                1. 不允许多个同名的强符号
                2. 强符号和弱符号同名，选择强符号
                3. 多个弱符号同名，任意选择一个
            - static library:
                - Linux中archive
                - E-可重定位目标文件集合，U-未解析符号集合，D-已定义符号集合
                - *.a位置很重要
    - .rel.text：
        - .text节中的位置列表，链接时修改位置
        - 任何调用外部函数或者引用全局变量的指令
        - 可执行目标文件中通常省略
    - .rel.data:
        - 被引用或定义的所有全局变量的重定位信息
    - .debug:
        - 局部变量和类型定义、定义和引用的全局变量、原始C源文件
        - 以-g选项调用才生成
    - .line:
        - 源文件中行号与.text中机器指令的映射
        - 以-g选项调用才生成
    - .strtab:
        - .symtab和.debug中的符号表
        - section header中的section name

    - Section header table:
        - 描述了不同section的位置和大小
        - 每个section有固定大小的entry
- Relocation:
    - Entries:(.rel.text & .rel.data)
        - offset: 要被修改的重定位处的节偏移
        - type:
            - R_X86_64_PC32: PC相对地址，PC指向下一条指令
            - R_X86_64_32： 绝对寻址
        - symbol: 某全局变量或者函数
        - addend: 偏移调整
    - 算法：
        - 首先： refptr = s + r.offset（要被修改的重定位位置）
        - PC相对引用：
            1. refaddr = ADDR(s) + r.offset（要被修改的地方的运行时地址 = 当前section的运行时地址 + 偏移量）
            2. *refptr = (unsigned)(ADDR(r.symbol) + r.addend - refaddr)（把要被修改的地方的值改为运行时相对地址）
        - 绝对引用：
            1. *refptr = (unsigned)(ADDR(r.symbol) + r.addend - refaddr) （把要被修改的地方的值改为绝对地址）
- Executable Object Files
    - 文件格式：
        1. ELF头还包括了程序入口点（entry point）
        2. .text,.rodata和.data与可重定位目标文件相似
        3. .init定义了函数_init，程序初始化代码调用
        4. 不再需要.rel节
    - 对齐要求：vaddr mod align = off mod align（起始地址和节偏移量对齐）
    - 加载
- Dynamic Linking with Shared Libraries
    - 目标： 解决static library的缺陷（显式更新，浪费内存）
    - Linux系统接口：

```
#include <dlfcn.h>

//成功返回指向句柄的指针，出错返回NULL
//filename: ./*.so
//flag: RTLD_NOW(立即解析对外部符号的引用)，RTLD_LAZY(推迟符号解析直到执行来自库中的代码)
void *dlopen(const char *filename, int flag);

//成功返回指向符号的指针，出错返回NULL
void *dlsym(void *handle, char *symbol);

//成功返回0，出错返回-1
int dlclose(void *handle);
```

```
//对前面三者调用失败返回错误消息，调用成功返回NULL
const char *dlerror(void);
```

    - 技术实现——PIC(Position-Independent Code)
        - GOT(Global Offset Table):数据段
            - 8 bytes per entry
            - GOT[0] GOT[1] 解析函数地址使用
            - GOT[2] ld-linux.so的入口点
            - GOT[3] 系统start up
            - 从GOT[4]开始指向PLT entry第二条指令或运行时地址
        - PLT(Procedure Linkage Tabl代码段
            - 16 bytes per entry
            - PLT[0] 跳转到dynamic linker
            - PLT[1] 调用系统启动函数__libc_start_main
            - 从PLT[2]开始调用用户代码函数
- Tools:
    - AR: 静态库
    - STRINGS：列出一个目标文件中所有可打印字符串
    - STRIP：从目标文件删除符号表信息
    - NM：列出一个目标文件的符号表中定义的符号
    - SIZE：列出目标文件中节的名字和大小
    - READELF：显示一个目标文件的完整结构（包含SIZE和NM）
    - OBJDUMP：所有二进制工具之母。反汇编.text