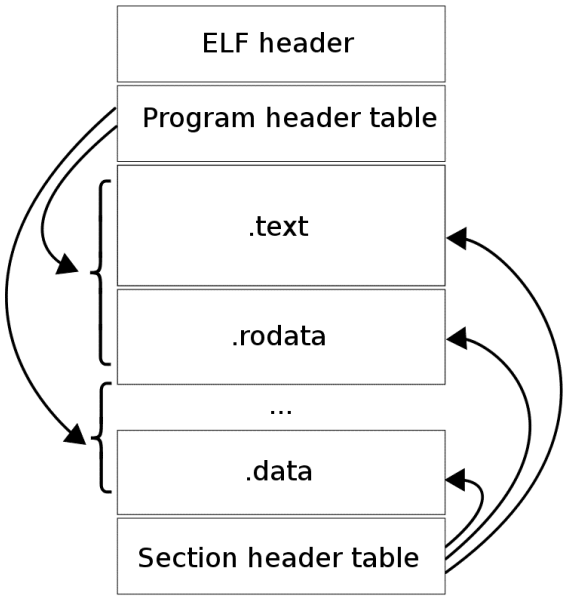
Tuần 4: Tìm hiểu về ELF Format

Sơ lược về định dạng ELF:
Định dạng ELF được sử dụng rỗng rãi cho executable files, relocatable object files, shared libraries, và core dumps. Vào năm 1999, dự án 86open đã lựa chọn ELF là định dạng tệp tin nhị phân tiêu chuẩn trên bộ xử lý x86. Từ đó tới nay, ELF được sử dụng trong một vài hệ điều hành khác. Trong số đó bao gồm Linux, Solaris/Illumos, BSD families (OpenBSD, FreeBSD, NetBSD,..), BeOS/Haiku, and Fuchsia OS (new Google's operating system). Hơn nữa, ELF còn xuất hiên trên cả các nền tảng di động như Android, Maemo, Sailfish OS hay các nền tảng game console như PlayStation, Wii.

Cấu trúc của ELF



1. ELF Header
ELF Header có độ dài 32 bytes, quy định các tính chất của file. ELF Header bắt đầu với 4 bytes đầu là 0x7f, 0x45, 0x4c và 0x46.

```
→  week4_asm xxd ./dtcntt | head -5
00000000: 7f45 4c46 0201 0100 0000 0000 0000 0000  .ELF............
00000010: 0300 3e00 0100 0000 6010 0000 0000 0000  ..>.....`.......
00000020: 4000 0000 0000 0000 c038 0000 0000 0000  @........8......
00000030: 0000 0000 4000 3800 0d00 4000 1e00 1d00  ....@.8...@.....
00000040: 0600 0000 0400 0000 4000 0000 0000 0000  ........@.......
→  week4_asm
```

Cấu trúc ELF Header

```
#define EI_NIDENT 16

typedef struct {
    unsigned char e_ident[EI_NIDENT];
    uint16_t      e_type;
    uint16_t      e_machine;
    uint32_t      e_version;
    ElfN_Addr     e_entry;
    ElfN_Off      e_phoff;
    ElfN_Off      e_shoff;
    uint32_t      e_flags;
    uint16_t      e_ehsize;
    uint16_t      e_phentsize;
    uint16_t      e_phnum;
    uint16_t      e_shentsize;
    uint16_t      e_shnum;
    uint16_t      e_shstrndx;
} ElfN_Ehdr;
```

ElfN với N là 32bit hay 64bit.

- e_ident gồm một dãy các bytes quy định thuộc tính file, độc lập với bộ xử lý.

    EI_MAG: 4 bytes 0x7f 0x45 0x4c 0x46

    EI_CLASS: quy định kiến trúc của file

    EI_DATA: quy định mã hóa dữ liệu

    EI_VERSION: quy định phiên bản của ELF

    EI_OSABI: quy định hệ điều hành

    EI_ABIVERSION: quy định phiên bản ABI

    EI_PAD: padding

    EI_NIDENT: quy định kích cỡ của mảng e_ident

- e_type: quy định kiểu file

- e_machine: quy định cấu trúc bắt buộc

- e_version: quy định phiên bản file

- e_entry:

- e_phoff: chứa program header’s offset
- e_shoff: chứa section header’s offset
- e_flags: chứa cờ của file
- e_ehsize: chứa kích thước của ELF Header
- e_phentsize: chứa kích thước của program header
- e_phnum: chứa số lượng mục trong program header
- e_shentsize: chứa kích thước của section header
- e_shnum: chứa số lượng mục trong section header
- e_shstrndx: chứa vị trí của mục section name trong file

2. Program Header

Program Header có kích thước 32 bytes với cấu trúc 32bit và 56 bytes với cấu trúc 64bit. Program Header cần thiết cho hệ thống để chuẩn bị thực thi file.

Cấu trúc program header

```
typedef struct {

        uint32_t   p_type;
        Elf32_Off  p_offset;
        Elf32_Addr p_vaddr;
        Elf32_Addr p_paddr;
        uint32_t   p_filesz;
        uint32_t   p_memsz;
        uint32_t   p_flags;
        uint32_t   p_align;
    } Elf32_Phdr;
```

```
    typedef struct {
        uint32_t   p_type;
        uint32_t   p_flags;
        Elf64_Off  p_offset;
        Elf64_Addr p_vaddr;
        Elf64_Addr p_paddr;
        uint64_t   p_filesz;
        uint64_t   p_memsz;
        uint64_t   p_align;
    } Elf64_Phdr;
```

- p_type: quy định loại của đoạn phần tử mảng đang mô tả

- p_offset: chứa offset của đoạn

- p_vaddr: chứa virtual address của đoạn

- p_paddr: chứa physical address nếu ở trên hệ thống có yêu cầu physical addressing. Ở BSD, trường giá trị này bằng 0.

- p_filesz: chứa kích thước tính theo bytes trong file của đoạn

- p_memsz: chứa kích thước tính theo bytes trong memory của đoạn.

- p_flags: chứa cờ của đoạn

- p_align: chứa giá trị cho biết segment cần điều chỉnh. p_align là 0 và 1 nghĩa là không cần căn chỉnh. Còn lại, p_align là số dương và p_vaddr = p_offset module p_align.

3. Section Header

Section Header Table xác định vị trí của tất cả các thành phần của file. Section Header có kích thước 0x28 bytes với cấu trúc 32bit và 0x40 bytes với cấu trúc 64bit.

Cấu trúc Section Header

```
typedef struct {
   uint32_t   sh_name;
   uint32_t   sh_type;
   uint32_t   sh_flags;
   Elf32_Addr sh_addr;
   Elf32_Off  sh_offset;
   uint32_t   sh_size;
   uint32_t   sh_link;
   uint32_t   sh_info;
   uint32_t   sh_addralign;
   uint32_t   sh_entsize;
} Elf32_Shdr;

typedef struct {
   uint32_t   sh_name;
   uint32_t   sh_type;
   uint64_t   sh_flags;
   Elf64_Addr sh_addr;
   Elf64_Off  sh_offset;
   uint64_t   sh_size;
   uint32_t   sh_link;
   uint32_t   sh_info;
   uint64_t   sh_addralign;
   uint64_t   sh_entsize;
} Elf64_Shdr;
```

- sh_name: chứa vị trí bắt đầu của tên section trong “section header string table” .shstrtab

- sh_type: chứa loại của section

- sh_flags: chứa cờ của section

- sh_addr: chứa virtual address của section được nạp vào memory

- sh_offset: chứa offset của section trong file

- sh_size: chứa kích thước của section trong file

- sh_link: chứa index của một section header table tùy thuộc vào type của section

- sh_info: chứa trông tin thêm của section

- sh_addralign: chứa align bắt buộc của section

- sh_entsize: chứa kích thước tính theo bytes của section