Trabalho 1 - Desmontador

Seu objetivo neste trabalho é implementar uma versão simplificada da ferramenta *objdump* para executáveis ELF compilados para a linguagem de máquina do RISC-V.

Você pode ver todas as opções e recursos da ferramenta *objdump* do seu sistema utilizando o comando `man llvm-objdump-13`. Neste trabalho, você deverá implementar um programa que reproduz as saídas das opções "`-h`", "`-t`" e "`-d`". Elas imprimem, respectivamente, a tabela de seções, a tabela de símbolos, e o código em linguagem de montagem da seção ".text". A entrada e saída de seu programa deverão ser exatamente as mesmas do `llvm-objdump-13` para os comandos descritos. Vale notar que essas são as mesmas informações que você coletou manualmente do *hexdump* de um arquivo ELF no laboratório 4. Seu trabalho agora é apenas automatizar esta coleta.

Instruções:

- Seu código deverá ser escrito inteiramente na linguagem C e compilado para x86 com o seguinte comando:
  - `gcc desmontador.c -Wall -Werror -g -o desmontador`
- Seu código deve estar integralmente contido em um único arquivo chamado `desmontador.c`
- As únicas funções externas permitidas são:
  - Função `open` da biblioteca fcntl.h
  - Funções `read`, `write` e `close` da biblioteca unistd.h
- **Você NÃO pode usar quer outra biblioteca (incluindo `stdio`, `stdlib` e `string`) ou função externa em seu trabalho**. O uso de qualquer outra biblioteca (incluindo `stdio`, `stdlib` e `string`) ou função externa implicará em **nota 0** no trabalho.
- Para o comando "`-d`", apenas as instruções listadas na Figura 2.3 do Capítulo 2 do Guia Prático RISC-V: Atlas de uma Arquitetura Aberta devem ser consideradas.
  - Caso a instrução a ser decodificada não esteja na tabela, seu objdump deve emitir `<unknown>` no lugar do nome da instrução.
  - O objdump não deve gerar pseudo-instruções. Para isso, adicionamos a opção "`-M=no-aliases`" logo após o comando `llvm-objdump-13`.
- Seu programa receberá como entrada apenas executáveis compilados com os comandos apresentados no laboratório 1. Os executáveis terão no máximo 100KB.
- Para facilitar a correção automática, nós substituiremos todos os espaços consecutivos da impressão do Objdump por um único espaço.

Exemplos de entrada e saída:

Assim como o `llvm-objdump-13`, seu programa receberá a entrada através dos argumentos da linha de comando. Exemplo de execuções válidas para um executável "test.x":

- `./desmontador -d  test.x`

- `./desmontador -t  test.x`
- `./desmontador -h  test.x`

Para ver a saída esperada em cada caso:

1. Execute os seguintes comandos sobre o executável:
    - `llvm-objdump-13 -M=no-aliases -d test.x`
    - `llvm-objdump-13 -t test.x`
    - `llvm-objdump-13 -h test.x`

Entrega e Avaliação

Você deve submeter o arquivo `desmontador.c` no moodle.
Para cada executável testado, uma nota será atribuída da forma mostrada a seguir. Não há nota parcial dentro de cada um dos itens.

- 20% para a impressão correta da tabela de símbolos (opção `-t`).
- 20% para a impressão correta da tabela de seções (opção `-h`). A coluna *type* não será considerada para a correção.
- 20% para a impressão correta da desmontagem da seção `.text` (opção `-d`) até a primeira coluna.
- 20% para a impressão correta da desmontagem da seção `.text` (opção `-d`) até a segunda coluna.
- 20% para a impressão correta da desmontagem da seção `.text` (opção `-d`) até a terceira coluna.

Seu código será testado automaticamente por um script para N executáveis diferentes. Sua nota final no trabalho é a média simples entre as notas dos N testes.

Dicas

- Atente-se a versão da ferramenta `llvm-objdump`, estamos utilizando a versão 13.
    - Alguns computadores possuem apenas o comando `llvm-objdump`, em vez do comando `llvm-objdump-13`. Neste caso, execute o comando com a *flag* `--version` e verifique se a mesma produz a informação `LLVM version 13.0.0`. Caso positivo, você pode usar este comando.
- A versão 13 do llvm-objdump não reconhece a opção -m=no-aliases. Neste caso, utilize -M no-aliases.
- As funções implementadas no Laboratório 3 podem te ajudar a imprimir a saída.
- Todos os arquivos a serem desmontados são compilados para RISCV 32 *bits* e *little-endian*.
- No Laboratório 4, você interpretou um ELF a partir do seu *hexdump* (O *hexdump* é apenas a impressão *byte* a *byte* do arquivo). Para depurar seu programa, pode ser útil visualizar o *hexdump* dos ELFs fornecidos. Para isso, utilize o comando `hexdump -C executavel.x`
- Caso tenha dúvidas sobre como trabalhar com argumentos passados pela linha de comando, veja: C argc and argv Examples to Parse Command Line Arguments

- O comando *readelf* também pode te ajudar a visualizar informações sobre cada ELF. Utilize-o da seguinte forma: `readelf -a executavel.x`
- Para identificar a qual seção cada símbolo pertence, observe o campo `st_shndx` da `symtab` (ele indica o index da seção do rótulo). A estrutura de cada entrada da `symtab` está descrita pela *struct* `Elf32_Sym` neste documento. Os tipos `Elf32_Word`, `Elf32_Addr` e `Elf32_Half` possuem, respectivamente, 4, 4 e 2 *bytes*.

Prazo de Entrega:

- ~~Prazo 1 (valendo 100% da nota):~~
  - ~~**até 27/09 às 13h59**~~
- ~~Prazo 2 - Entrega atrasada (valendo 70% da nota) :~~
  - ~~**de 27/09 às 14h00 até 28/9 às 13h59**~~
  - ~~Neste caso, a nota será multiplicada por **0.7**.~~
- ~~Não serão aceitas submissões após às 13h59 do dia 28/9.~~
- Prazo 1 (valendo 100% da nota):
  - **até 28/9 às 23h59**
- Prazo 2 - Entrega atrasada (fator multiplicativo = 0.8):
  - **de 29/9 00h00 até 4/10 às 13h59**
- Prazo 3 - Entrega atrasada (fator multiplicativo = 0.6):
  - **de 4/10 às 13h59 até 5/10 às 13:59**
- Não serão aceitas submissões após às 13h59 do dia 5/10