Guião 4

Objetivos:

- Identificar e corrigir um erro de *buffer overflow*;
- Usar aritmética de apontadores para aceder a elementos do *array*;
- Explorar endereços de *stack*;
- Usar *Reverse Execution* no GDB.

Buffer

1. Altere o programa para executar apenas a função **buffer0**
2. Compile o programa usando **gcc -g -O0 main.c -o buffer**
3. Por agora, ignore os *warnings*. Analise as *flags* de compilação
4. Execute o programa e teste com as *strings* **HELLO** e **HELLOWORLD**
   O que sucede em cada um dos casos e porquê?
5. Execute o programa usando o **gdb**
6. Coloque um **breakpoint** no buffer e execute o programa
7. Verifique o comportamento do programa no gdb com as *strings* anteriores (utilize os comandos de análise passo a passo)
8. Corrija o programa de modo a evitar o problema

Apontadores

1. Altere o programa para executar apenas a função **pointersAri**
2. Compile o programa usando **gcc -g -O0 main.c -o pointers**
3. Por agora, ignore os *warnings*
4. Execute o programa no **gdb**
5. Coloque um **breakpoint** na linha que imprime dados e execute o programa
6. Imprima o valor das variáveis **n**, **ptr** e **i** antes e depois de executar a instrução do **printf**
7. Modifique o programa de forma a imprimir todos os valores do *array* e respetivo endereço de memória associada a cada elemento
8. Modifique o programa de forma a imprimir um *array* com os valores acumulados e respetivo endereço de memória associada a cada elemento.
   Exemplo: os acumulados de [10, 20, 30, 40, 50] resulta em [10, 30, 60, 100, 150]

Endereços de stack

1. Altere o programa para executar apenas a função **addressSt**
2. Compile o programa usando **gcc -g -O0 main.c -o address**
   Porque é que surge o *warning*?
3. Execute o programa e explique o resultado obtido
4. Execute o programa no **gdb**
5. Analise o erro e corrija o programa.

Reverse debugging

1. Altere o programa para executar apenas a função **arrayR**
2. Compile o programa usando **gcc -g -O0 main.c -o array**

3. Execute o programa e explique o resultado obtido
4. Abra o programa no **gdb**
5. Vamos usar **reverse debugging** para analisar o problema. Aceda a https://sourceware.org/gdb/current/onlinedocs/gdb.html/Reverse-Execution.html para saber mais sobre o tema
6. Execute um programa usando **start**
7. Ative o registo de execução para que o gdb grave os estados e permita fazer o *reverse debugging* (comando **record**)
8. Coloque um **breakpoint** na instrução do *printf*
9. Execute o programa e examine o estado do *array*
10. Use a execução inversa para analisar o conteúdo do *array* e analisar o que acontece (comandos **reverse-next** e **reverse-step**)
11. Verifique o valor da variável **i**. Sugere-se usar os comandos **watch** e **where**. O que sucede com a mesma?
12. Corrija o código para evitar o erro.