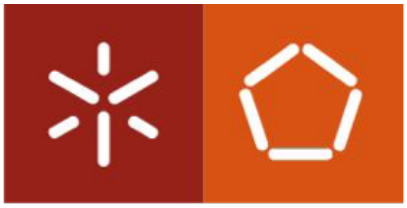

Universidade do Minho
Departamento de Informática
Mestrado Integrado em Engenharia Informática

Engenharia de Segurança

Aula 11

Diana Lopes, nº a74944
Gabriela Vaz, nº a74899

Conteúdo

Capítulo 1

Pergunta P1.1

1. A vulnerabilidade está relacionada com o tamanho dos argumentos x e y da função vulneravel(). Quando os valores de x e y são demasiado grandes corre-se o risco de começar a escrever em posições de memória não alocadas para o efeito.

2.
```
int main(){
    char *m;
    size_t x = 14729562724;
    size_t y = 284938574723;
    char valor = 'a';
    vulneravel(m,x, y, valor);
    return 0;
}
```

3. Ao compilar o programa, dá *Segmentation fault*. Isto acontece devido à vulnerabilidade encontrada na alínea 1 desta pergunta.

Capítulo 2

Pergunta P1.2

1. Ao contrário do que acontece na questão anterior, neste programa limita-se o tamanho máximo das variáveis. No entanto, não se estabelece um tamanho mínimo. Ora, se o tamanho do input for o valor 0 a variável **tamanho_real** vai assumir o valor -1.

2.
```
int main(){
    char *o;
    size_t tamanho=0;
    vulneravel(o,tamanho);
    return 0;
}
```

3. À semelhança da questão anterior, quando se executa o programa dá *Segmentation fault.*

Capítulo 3

Pergunta P1.3

1. A variável **tamanho_real** é do tipo **int**. O intervalo de valores contemplado pelo tipo de dados **int** é menor do que o intervalo de valores contemplado pelo tipo de dados **size_t**. Por essa razão, a vulnerabilidade existente é semelhante à da questão 1.

2.
```
int main(){
    char *o;
    size_t tamanho=1024327592302;
    vulneravel(o,tamanho);
    return 0;
}
```

3. À semelhança das questões anteriores, quando se executa o programa dá *Segmentation fault*.