

Universidade do Minho

Mestrado Integrado em Engenharia Informática

Engenharia de Segurança

Trabalho TP9

Grupo 2

Paulo Gameiro - A72067 Pedro Rodrigues - PG41092 Rafaela Soares - A79034

Braga, Portugal 11 de Maio de 2020

Conteúdo

1	1 Pergunta 1.1	
	1.1 1.1.1	
	$1.2 1.1.2 \dots \dots \dots \dots$	
	1.3 1.1.3	
2	2 Pergunta 1.2	:

1 Pergunta 1.1

$1.1 \quad 1.1.1$

Duas das vulnerabilidades que podem ser evidenciadas no ficheiro filetype.c tratam-se da Meta-caracteres e *Out-of-bounds Read*.

No que diz respeito à vulnerabilidade Metacaracteres, esta é do tipo injeção de separadores. Como a função system permite executar programas no terminal e não existe verificação da *string* colocada como *input* pelo utilizador, é possível que esta vulnerabilidade seja explorada, o que não é o desejável.

No que concerne à vulnerabilidade *Out-of-bounds Read*, esta pode ser explorada, uma vez que não é averiguado se o *input* argv[1] tem o tamanho expectável, neste caso, de 64 caracteres. Para além disso, também não é apurado se o número de parâmetros colocado pelo utilizador como *input* é o pretendido.

1.2 1.1.2

Para se explorar a vulnerabilidade Metacaracteres, através da função system, basta colocar o metacaractere ';' seguido de um comando simples como "ls", por exemplo.

```
cosmicgirl@cosmicgirl-VirtualBox: ~/Desktop

Ficheiro Editar Ver Procurar Terminal Ajuda

cosmicgirl@cosmicgirl-VirtualBox: ~/Desktop$ ./filetype oi.txt; ls

oi.txt: UTF-8 Unicode text
2.y '[PL] TP1_versaoAnterior'
clang+llvm-10.0.0-x86_64-linux-gnu-ubuntu-18.04 '[PL] treino'
'def validaNumero():.y' SPARTA-master
filetype
filetype.c '[TP1] Grupo56'
lisp.y '[TP1] Grupo56-Zip'
listaCompras.y '[TP1] Grupo56-Zip'
tstat-3.1.1
valida.py
'cosmicgirl@cosmicgirl-VirtualBox: ~/Desktop$
```

Figura 1: Exemplo de exploração da vulnerabilidade Metacaracteres

Já para se explorar a vulnerabilidade *Out-of-bounds Read*, bastava colocar como parâmetro um ficheiro em que o seu nome tenha mais de 64 caracteres, de forma a que o programa não tenha sucesso na execução. Tal acontece, pois não é lido todo o nome do ficheiro.

Figura 2: Exemplo de exploração da vulnerabilidade Out-of-bounds Read

1.3 1.1.3

As permissões setuid root permitem que qualquer utilizador possa executar um programa, com os privilégios do proprietário desse programa. Tal permite que este tenha acesso a funções que normalmente não teria.

Se o presente programa no ficheiro filetype.c tivesse permissões setuid root, a segurança deste estaria consideravelmente em causa. Como o programa não implementa qualquer tipo de verificação no input colocado na função system, o utilizador poderia executar o que pretendesse, com cariz malicioso.

2 Pergunta 1.2

A proposta de solução desta pergunta realizada na linguagem Python, o ficheiro valida.py, encontra-se disponibilizada no GitHub do grupo.

Este é um programa que solicita ao utilizador que insira parâmetros como valor a pagar, nome, data de nascimento, NIF, NIC, número de cartão de crédito, validade do cartão de crédito e o CVC associado ao mesmo. Caso este não insira, conforme o pretendido, não pode avançar para o próximo campo.

As condições necessárias para a validação de cada *input* e a respetiva função correspondente elaborada são as seguintes:

- Valor a pagar : Este tem de ser constituído por um máximo de 9 dígitos. São apenas permitidos valores maiores que 0 e valores decimais, apenas se o formato corresponder a _.02 ou _.20.
- **Nome**: Este tem de começar por uma letra e deve ter no mínimo 7 letras e no máximo 50. De referir que só serão permitidos *inputs* que sejam constituídos por palavras e espaços entre palavras. A função que valida este campo é a validaNome.
- Data de Nascimento: Tem de estar de acordo com o formato %YYYY-%M-%D e ser anterior à data atual. A função que valida este campo é a validaData.
- NIF: Este tem de ter 9 dígitos. A função que valida este campo é a validaNif_Nic.
- NIC: Este tem de ter 9 dígitos. A função que valida este campo é a validaNif_Nic.
- Número de cartão de crédito: Este tem de ter entre 7 a 19 dígitos. A função que valida este campo é a validaNrCartao.
- Validade do cartão de crédito: Tem de estar de acordo com o formato %YYYY-%M-%D e ser posterior à data atual. A função que valida este campo é a validaData.
- CVC: Este tem de ser constituído por 3 dígitos. A função que valida este campo é a validaCVC.