

17-02-2025

Riscos de segurança em aplicações web

Relatório do Trabalho 1

Diana Fernandes e João Almeida
UNIVERSIDADE DE AVEIRO

Índice

Introdução.....	2
Desenvolvimento	3
Conclusão.....	8

Introdução

Com base na unidade curricular de Redes e Segurança Informática, desenvolvemos em grupo um trabalho relacionado a vários tipos de ficheiro, como XML; PYTHON e JSON, com o objetivo de armazenar informações de produtos.

Também foi visto várias sugestões para minimizar e prevenir os riscos de segurança.

Desenvolvimento

1. Desenvolva um ficheiro XML que permita guardar a informação de 2 produtos. (nome, preço, imagem, categoria)

Na questão 1, começamos por criar um ficheiro XML, com o nome "Produtos", para armazenar a informação de dois produtos. Este ficheiro, contém elementos para o nome, preço, imagem e categoria de cada produto.

```
<?xml version="1.0" encoding="UTF-8"?>

<produtos>
  <produto1>
    <nome>HP by Victus</nome>
    <preço>1200.00</preço>
    <imagem>hp.jpg</imagem>
    <categoria>computador</categoria>
  </produto1>

  <produto2>
    <nome>Portátil MSI</nome>
    <preço>1599.90</preço>
    <imagem>katana.jpg</imagem>
    <categoria>computador</categoria>
  </produto2>
</produtos>
```

2. Desenvolva um programa em python para ler o ficheiro XML e imprimir na consola.

Sugestão: <https://docs.python.org/3/library/xml.etree.elementtree.html>

Na questão 2, criamos um ficheiro em python, com o nome "ler_produtos.py", com o objetivo em ler o ficheiro XML e imprimir na consola a informação dos produtos.

```
import xml.etree.ElementTree as ET
tree = ET.parse('produtos.xml')
root = tree.getroot()

#print(root.tag)
#print(root.attrib)

titulo = "PRODUTOS"
print(f"{titulo:^30}")

for child in root:
    print(child[0].tag, "-", child[0].text)
    print(f"{child[1].tag} = {child[1].text}€")
```

```
print(child[2].tag, "-", child[2].text)
print(child[3].tag, "-", child[3].text)
print()
```

O código utiliza uma biblioteca para ler o XML ("xml.etree.ElementTree")

3. Modifique o ficheiro XML de forma que a informação da categoria dos produtos seja através de entidades internas.

Sugestão: https://www.w3schools.com/xml/xml_dtd_entities.asp

Na questão 3, voltamos para o ficheiro XML, "Produtos", em que no topo modificamos o ficheiro e incluímos as entidades internas para representar as categorias dos produtos.

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE CATEGORIA [
    <!ENTITY categoria "computador">
<!-- <!ENTITY categoria SYSTEM "file:///etc/passwd"> -->
]>

<produtos>
    <produto1>
        <nome>HP by Victus</nome>
        <preço>1200.00</preço>
        <imagem>hp.jpg</imagem>
        <categoria>computador</categoria>
    </produto1>

    <produto2>
        <nome>Portátil MSI</nome>
        <preço>1599.90</preço>
        <imagem>katana.jpg</imagem>
        <categoria>computador</categoria>
    </produto2>
</produtos>
```

4. Confirme que o programa em python imprime a informação correta dos produtos, nomeadamente da categoria.

Na questão 4, o programa em python desenvolvido na questão 2, continuará a funcionar corretamente após a modificação do XML, porque esta biblioteca já resolve automaticamente as entidades definidas.

O resultado é este:

```
PS Microsoft.PowerShell.Core\FileSystem::\\arca.ua.pt\utilizadores\dianacfernandes@ua.pt\I
fernandes@ua.pt\Desktop\Redes e Segurança Informática\Aula1\ler_produtos.py"
    PRODUTOS
nome - HP by Victus
preço = 1200.00€
imagem - hp.jpg
categoria - computador

nome - Portátil MSI
preço = 1599.90€
imagem - katana.jpg
categoria - computador

PS Microsoft.PowerShell.Core\FileSystem::\\arca.ua.pt\utilizadores\dianacfernandes@ua.pt\I
```

5. Verifique se o programa em python apresenta riscos de segurança associados a ficheiros XML, como:
- suporte a entidades externas;
 - billion laughs.

Na questão 5, criámos um ficheiro XML, com o nome "Bomb", com o objetivo de testar o programa para prevenir ataques com o intuito de ler ficheiros sensíveis no programa.

O "Billion Laughs" é um ataque de negação do serviço(tipo spam), o objetivo destes ataques é fazer o uso excessivo de memória para criar uma abertura ou enfraquecer o programa.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE xmlbomb [
  <!ENTITY a 'omg'>
  <!ENTITY b "&a;&a;&a;&a;&a;&a;&a;">
  <!ENTITY c "&b;&b;&b;&b;&b;&b;&b;">
  <!ENTITY d "&c;&c;&c;&c;&c;&c;&c;">
  <!ENTITY e "&d;&d;&d;&d;&d;&d;&d;">
]>

<bomb>&e;</bomb>
```

```
import xml.etree.ElementTree as ET
tree = ET.parse('bomb.xml')
root = tree.getroot()
```

```
print(root.text)
```

6. Sugestões para resolver ou minimizar estes riscos de segurança?

Para prevenir ou minimizar estes riscos de segurança são:

- Prevenir o Billion Laughs Attack: Limitar o tamanho de entidades no XML;
- Usar bibliotecas seguras: por exemplo, lxml ou defusedxml;
- Desabilitar Entidades Externas: deve-se desabilitar o suporte a entidades externas.

7. Desenvolva um ficheiro JSON que permita guardar a mesma informação da alínea 1)

Considere os exemplos em http://www.w3schools.com/js/js_json_xml.asp

Na questão 7, criámos um ficheiro JSON, com o nome "produtos.json", para armazenar as mesmas informações dos produtos.

```
{
  "produtos": [
    {
      "nome": "HP by Victus",
      "preco": 1200.00,
      "imagem": "hp.jpg",
      "categoria": "computador"
    },
    {
      "nome": "Portatil MSI",
      "preco": 1599.90,
      "imagem": "katana.jpg",
      "categoria": "computador"
    }
  ]
}
```

8. Desenvolva um programa em python para ler o JSON e imprimir na consola.

Na questão 8, desenvolvemos um ficheiro em python, com o nome "ler_json.py", para ler e imprimir a informação dos produtos.

```
import json

f = open("produtos.json", "r")
data = f.read()
f.close()

print(type(data)) #str
#print(data)

data = json.loads(data) #se precisar de converter para dicionário
print(type(data)) #dict
print(json.dumps(data, indent = 4))
```

9. Desenvolva um programa em python que solicite dados ao utilizador, guarde num dicionário, e depois converta para JSON.

Por fim, na última questão, criamos um ficheiro, com o nome "ex9_json.py", que solicita ao utilizador os dados, depois armazena as informações num dicionário e converte para JSON.

```
import json
dados = {}

while True:
    opcao = input("Pretende adicionar informação: ")
    if opcao[0].lower() in ["s"]:
        campo = input("campo: ")
        valor = input("valor: ")
        dados[campo]=valor
    elif opcao[0].lower() == "n":
        break

#imprimir
dados_json = json.dumps(dados, indent=2)
print(dados_json)

#gravar num ficheiro dados.json
f = open("dados.json", "w")
f.write(json.dumps(dados, indent=2))
f.close()
```


Conclusão

Neste relatório, abordamos a criação de ficheiros XML, JSON para armazenar informações de produtos, desenvolvemos de programas em PYTHON para ler e imprimir esses ficheiros. A análise de riscos de segurança associados ao processamento de XML e sugestões para minimizar esses riscos.

Além disso, foi também desenvolvido um programa para solicitar dados ao utilizador e convertê-los para o formato JSON.

Este trabalho foi muito enriquecedor, pois aprendemos bastantes coisas ao qual será usado no nosso dia a dia.