Questões de Segurança em Engenharia de Software (QSES), 2018/19
Mestrado em Segurança Informática
Departamento de Ciência de Computadores
Faculdade de Ciências da Universidade do Porto
Exame de época normal – 15/01/2019
Duração: 2:30

Grupo A

Considere o Fragmento 1 de código Java abaixo, adaptado da aplicação Java Vulnerable Lab. O código relaciona-se com a apresentação do conteúdo de uma entrada na tabela de mensagens **`POSTS`** em correspondência ao "post" identificado pelo parâmetro **`post_id`**. Deve assumir que o valor do parâmetro **`post_id`** pode ser malicioso e que o conteúdo da tabela **`POSTS`** poderá também não ser seguro.

Fragmento de código 1:

```
String post_id = request.getParameter("post_id");
out.println("Post ID: " + post_id + "<br/>");
try {
  Connection db = ... ;
  Statement stmt = db.createStatement();
  ResultSet rs = stmt.executeQuery("SELECT * FROM POSTS WHERE ID=" + post_id);
  out.println("Title: "      + rs.getString("title")   + "<br/>");
  out.println("Posted by: "  + rs.getString("user")    + "<br/>");
  out.println("Content: "    + rs.getString("content") + "<br/>");
}
catch (SQLException e) {
  out.println("Database error!<br/>");
  e.printStackTrace(out);
}
```

Questões:

1. [5.5 valores] Identifique as vulnerabilidades no fragmento de código, explicando para cada uma:

- o tipo de vulnerabilidade e de que forma pode ser explorada, dando exemplo de valores maliciosos para o parâmetro **`post_id`** ou para o conteúdo da tabela **`POSTS`**;
- as alterações necessárias ao código para eliminá-la — pode usar pseudo-código aproximado se não se lembrar de alguns detalhes, desde que o significado seja claro.

2. [2 valores] Explique em que consistem ataques do tipo "session hi-jacking" relacionados com o uso de "cookies" e 3 medidas no manuseamento de "cookies" que mitiguem este tipo de ataques.

Grupo B

Considere o Fragmento 2 de código em C abaixo. A função **processData** lê um identificador de dados **id** e usa-o para processar um ficheiro de dados **id.txt** na pasta **PATH_FOR_FILES**. Assuma que **PATH_FOR_FILES** é uma string constante definida que pode ter um tamanho arbitrário.

Fragmento de código 2:

```
void processData(void) {
  char id[32];  // id
  char* path;   // path de um ficheiro
  FILE* file;   // ficheiro
  gets(id);     // lê identificador
  path = (char*) malloc(128); // aloca memória para path do ficheiro
  sprintf(path, "%s/%s.txt", PATH_FOR_FILES, id); // define path do ficheiro
  if (access(path, R_OK) == 0) {   // testa se ficheiro pode ser lido
    FILE* file = fopen(path, "r"); // abre ficheiro para leitura
    ...                            // processa ficheiro de alguma forma
    fclose(file);                  // fecha ficheiro
  }  else {
    printf(path);                            // imprime path do ficheiro
    printf(": file cannot be read!\n");      // imprime causa de erro
  }
  free(path);                                // liberta memória alocada
  printf("Done processing %s\n", path);      // imprime mensagem final
}
```

Questões:

1. [5.5 valores] Identifique as vulnerabilidades no fragmento de código, explicando para cada uma:

- o tipo da vulnerabilidade e o seu possível efeito durante a execução;
- as alterações necessárias ao código para eliminá-la.

2. [2 valores] Explique os princípio básicos para materializar um "stack-smashing attack" no código e respectivas assunções. Indique e explique sucintamente também três mecanismos de protecção contra um ataque deste tipo.

Grupo C

Dê respostas claras e sucintas às seguintes perguntas.

1. [2 valores] Em relação ao uso de "fuzz testing":

- Qual é o objectivo geral?
- Indique três técnicas base empregues na geração de valores.
- Qual é a distinção entre as variantes "black-box" e "white-box" de "fuzz testing" ?

2. [1 valores] Indique uma vantagem e uma desvantagem do uso de ferramentas de análise estática face a ferramentas de "pen-testing" para validação da segurança de uma aplicação.

3. [1 valores] Em que consistem vulnerabilidades baseadas em "race conditions"?

4. [1 valores] Quais são as ideias e objectivos gerais do "security touchpoint model"?