Universidade da Beira Interior
Faculdade de Engenharia
Departamento de Informática

Segurança de Sistemas Informáticos
Guia para Aula Laboratorial 9
2º Ciclo em Engenharia Informática
2º Ciclo em Eng. Eletrotécnica e de Computadores
2º Ciclo em Matemática e Aplicações

Computer Systems Security
Guide for Laboratory Class 9
M.Sc. in Computer Science and Engineering
M.Sc. in Electrical and Computer Engineering
M.Sc. in Mathematics and Applications

Sumário
Instalação de um servidor *HyperText Transfer Protocol* (HTTP) com tecnologia de pré-processamento de hipertexto (*Hypertext Preprocessor* (PHP)) e do sistema de gestão de bases de dados MySQL. Elaboração de uma pequena aplicação *web* para teste de um simples vetor de ataque *Structured Query Language* (SQL) *injection* (SQLi). Elaboração de uma aplicação *web* suscetível a ataques de *Cross Site Scripting* (XSS).

Summary
Instalation of an HyperText Transfer Protocol (HTTP) server with Hypertext Preprocessor (PHP) technology, and of the database management system MySQL. Development of a small web application for testing a simple Structured Query Language (SQL) injection attack vector. Elaboration of a web application susceptible to Cross Site Scripting (XSS) attacks.

Pré-requisitos:
Algumas tarefas enunciadas em baixo devem ser executadas num sistema com um servidor de páginas *Hypertext Markup Language* (HTML), robustecido com a tecnologia PHP e com ligação ao sistema de gestão de bases de dados relacional *MySQL*. Pressupõe-se o acesso a um sistema com estas condições preenchidas ou com a possibilidade de instalar o *software* necessário.

1 Instalação do Servidor HTTP, do PHP e do MySQL

Instalation of the HTTP Server, PHP and MySQL

Para criar uma aplicação *web* baseado em páginas HTTP que permita, entre outras funcionalidades, servir de intermediário entre o utilizador final e uma base de dados, é necessário primeiro instalar um servidor de ficheiros, e habilitá-lo com a tecnologia de suporte a páginas dinâmicas. Um servidor de páginas HTTP bastante popular atualmente e com código fonte aberto é o Apache (`http://www.apache.org/`). Uma das tecnologias de suporte a páginas dinâmicas com mais provas dadas é conhecida por *Hypertext Preprocessor* (PHP) (`http://www.php.net/`), e um dos sistemas de gestão de bases de dados relacional que melhor se integra com os sistemas antes apontados é o MySQL (`http://www.mysql.com/`). Existem implementações e pacotes de instalação para os vários sistemas operativos de utilização mais comum. É inclusivé possível encontrar combinações pré-configuradas de todas as tecnologias referidas em pacotes únicos, cuja designação acaba tipicamente com as letras AMPP.

Q1.: O que significam as várias letras do acrónimo antes referido?

☐ Apache HTTP Server	☐ MySQL
☐ Internet Information Services	☐ SQLite
☐ PHP	☐ Postgresql
☐ Perl	☐ Python

Uma dessas combinações / pacotes é designada por XAMPP (`http://www.apachefriends.org/en/xampp.html`), em que 'X' abrevia a palavra *cross*, o que significa que é um pacote disponível para as várias plataformas (sistemas operativos). Seguindo a mesma ordem de ideias, deduz-se que o pacote LAMP é especialmente vocacionado para sistemas Linux, o WAMP para sistemas Windows, e o MAMP para MAC.

Tarefa 1 ***Task 1***

Caso não tenha alguma das tecnologias mencionadas antes disponíveis na sua máquina, a sua primeira tarefa consiste na sua instalação.

Sistema Operativo Microsoft Windows

No sistema operativo Windows, a instalação segue os trâmites normais associados à instalação de uma aplicação: (i) *download* do pacote de instalação; (ii) execução do mesmo; e (iii), ajuste de alguns parâmetros via caixas de diálogo do assistente de instalação. Os vários serviços têm de ser ativados após instalação para que o o sistema funcione. A aplicação disponibiliza normalmente um painel de controlo que permite colocar os serviços a correr facilmente, embora o utilizador possa não ficar com a ideia exata das configurações que disponibilizam o seu sistema em rede, e que devem ser sujeitas a auditoria no desenvolvimento de aplicações seguras e robustas.

Sistema Operativo Fedora

No sistema operativo Fedora, a instalação do servidor Apache, do MySQL, do PHP e dos pacotes de interligação destas tecnologias passa pela emissão de uma sequência de comandos parecida com a que é mostrada a seguir:

```
> su
```

(palavra-passe do administrador de sistemas)

```
> dnf -y install httpd php mysql
mysql-server php-mysql
```

Nota: depois de instalar os vários pacotes, é necessário colocar os serviços a correr e verificar que a *firewall* permite ligações *Transmission Control Protocol* (TCP) na porta 80 (HTTP) e na porta 3306 (MySQL) – esta última pode não ser necessária para ligações locais. Para correr a aplicação de configuração da *firewall* em sistemas recentes, execute no terminal o comando seguinte e configure as duas portas:

```
> system-config-firewall-tui
```

Para colocar os serviços (http e mysql) a correr, emita os seguintes comandos no terminal:

```
> /sbin/service httpd start
```

```
> /sbin/service mysqld start
```

ou (dependendo do *init system*)

```
> systemctl start httpd
```

```
> systemctl start mysqld
```

No final, pode correr o comando `mysqladmin version status` para verificar o estado da instalação.

Sistema Operativo Ubuntu

Em Ubuntu, o mesmo pode ser conseguido usando comandos semelhantes aos seguintes (todos os comandos requerem direitos de administração):

```
> sudo apt-get install apache2
```

```
> sudo apt-get install mysql-client
mysql-server php5 php5-mysql
```

Não deve esquecer que os serviços têm de ser colocados em execução. Tal tarefa pode ser levada a cabo com os dois comandos seguintes:

```
> sudo /etc/init.d/apache2 restart
```

```
> sudo /etc/init.d/mysqld restart
```

Nota importante: depois de colocar o servidor HTTP em funcionamento, deve ser possível ver uma página de teste no endereço `http://localhost/` usando o *browser*. Deve consultar a documentação do servidor HTTP para saber em que diretoria deve colocar os ficheiros a serem servidos por ele.

Tarefa 2 ***Task 2***

Na primeira vez que colocar o MySQL a correr, este não deve ter configurada qualquer palavra-chave de acesso com o utilizador `root`, e terá outros artefactos que não são necessários numa instalação real do sistema. Corra o comando

```
> /usr/bin/mysql_secure_installation
```

no terminal e siga as instruções para (i) configurar uma nova palavra-chave de `root`, (ii) desativar a ligação via *root* remota, (iii) desativar contas anónimas e (iv) remover a base de dados de teste. **Nota:** deve sempre procurar um bom gerador de sequências aleatórias para gerar as suas palavras chave, e.g., use

```
> openssl rand -hex 10
```

para gerar uma boa palavra-chave com 10 bytes aleatórios no terminal. Contudo, para efeitos desta e das próximas aulas, configure a palavra-passe `mastersofenginf` para o utilizador `root`.

2 Criar uma Página *Web* de *Login*

Create a Login Web Page

Nesta parte do guia vamos tentar criar uma página com um formulário com campos para inserção de um *username* e de uma *password*. Caso a combinação fornecida por um utilizador for válida, então é-lhe permitido o acesso a uma outra página que apenas diz: `O BENFICA É O MAIOR!`.

Tarefa 3 *Task 3*

A primeira tarefa consiste em criar uma base de dados com a tabela `User` que irá conter, por sua vez, dois campos do tipo `VARCHAR(30)`, um para guardar *usernames* e outro para guardar *passwords*. A coluna relativa aos *usernames* será chave primária. A sequência de comandos que traduz esta tarefa é:

```
> mysql -u root -p
```
(inserir `mastersofenginf`)

```
mysql > CREATE DATABASE Aula9;
mysql > USE Aula9;
mysql > CREATE TABLE User(username VARCHAR(30)
PRIMARY KEY, password VARCHAR(30));
```

Aproveitamos imediatamente o facto de estarmos dentro do ambiente MySQL para inserir 1 utilizador válido:

```
mysql > INSERT INTO User VALUES("Steven",
"Seagal");
```

Deve verificar se a inserção foi bem sucedida, usando:

```
mysql > SELECT * FROM User;
```

No final, deve sair para a consola através da instrução:

```
mysql > quit
```

Tarefa 4 *Task 4*

A segunda tarefa consiste em criar a página `login.html` em `/var/www/html`:

```
> su
```
(palavra-passe do root)

```
> cd /var/www/html
> nano login.html
```

No conteúdo deste ficheiro deve estar:

```
<html>
<head><title> Login Page </title></head>
<body>
  <form name="INPUT" method="POST" action="
      validate.php">
    Username: <input type="text" name="
        username"><br />
    Password: <input type="text" name="
        password"><br />
    <input type="submit" value="Carrega
        Benfica!">
  </form>
</body>
</html>
```

Q2.: Qual a utilidade/funcionamento desta página?

- ☐ A página redireciona o utilizador para sí própria quando este insere um *username*.
- ☐ A página redireciona o utilizador para outra página quando este insere a *password*.
- ☐ A página redireciona o utilizador para outra página quando este carrega no botão `Carrega Benfica!`.

Nota: para sair do editor, faça `CTRL+X` e depois escreva `Y`.

Tarefa 5 *Task 5*

A página `validate.php` é o próximo alvo desta secção. A ideia é construir uma página em PHP que verifique se os valores introduzidos pelo utilizador na página elaborada na tarefa anterior coincidem com os que estão na base de dados. Caso coincidam, deve imprimir `O BENFICA É O MAIOR!`.

O conteúdo dessa página deve ser semelhante a:

```
<html>
<head><title> Validation </title></head>
<body>
<?php
  $con = mysqli_connect("localhost","root","
      mastersofenginf") or die("Error
      connecting to DB!");
  mysqli_select_db($con, "Aula9");
  $username = $_POST['username'];
  $password = $_POST['password'];
  $query = "SELECT COUNT(*) FROM User WHERE
      username = '$username' AND password = '
      $password'";
  echo "A query e a seguinte: <br />". $query
      . "<br /><br />";

  $result = mysqli_query($con, $query);
  $line = mysqli_fetch_array($result);
  if( $line[0] > 0 )
    echo "O Benfica is the best!";
  else
    echo "Username or Password not valid!";

  mysqli_close($con);
?>
</body>
</html>
```

Procure definir o que as linhas mais importantes do excerto de código anterior fazem:

linha 5: _____

_____

linha 6: _____

_____

linha 7: _____

_____

linha 9: _____

_____

linha 12: _____

_____

linha 13: _____

_____

linhas 14 a 17: _____

_____

_____

_____

Tarefa 6 *Task 6*

Teste a pequena aplicação *web* que criou com várias combinações *username* e *password* e verifique que tudo está a funcionar corretamente.

Q3.: Consegue ver a *query* que é submetida ao sistema?

☐ Sim, consigo. ☐ Não, não está a funcionar.

Q4.: O sistema reage como esperado a uma combinação correta *username*/*password*?

☐ Sim, impecável. ☐ Não.

Q5.: O sistema reage como esperado a uma combinação incorreta *username*/*password*?

☐ Sim, impecável. ☐ Não.

Tarefa 7 *Task 7*

Analise a *query* que é submetida ao MySQL e tente induzir uma forma de conseguir obter a mensagem `O Benfica é o melhor!` **sem** saber uma combinação válida de *username*/*password*. O segredo está em como enganar o sistema de gestão de base de dados e o código PHP...

Q6.: Como se resolve o problema de segurança explorado nesta tarefa?

☐ Chamando os bombeiros.

☐ Saneando os *inputs* que chegam ao PHP através das caixas de texto.

☐ Evitando que os *inputs* que chegam ao PHP através das caixas de texto sejam interpretados como parte do código SQL.

☐ Através da função `$input = mysqli_real_escape_string($input).`

☐ Este problema não pode ser resolvido.

3 Cross Site Scripting

Cross Site Scripting

As próximas tarefas lidam com linguagens de *scripting* com as quais pode ainda não ter tido contacto acentuado (e.g., *JavaScript*), pelo que se presume algum esforço de investigação pessoal nesses temas.

Tarefa 8 *Task 8*

Crie a página `ola.php` na raíz dos documentos disponibilizados pelo servidor *HyperText Transfer Protocol* (HTTP) com o conteúdo mostrado a seguir. No final, teste a página e peça a um(a) colega que aceda ao seu endereço com o seu *browser*, usando como *Uniform Resource Locator* (URL) um endereço parecido com `http://seu_ip/ola.php?nome="NomeDoOuDaColega"`.

```
<html>
  <head>
   <?php header("X-XSS-Protection: 0"); ?>
   <title>PHP Test</title>
  </head>
  <body>
  <br>
  <?php if(isset($_GET['nome'])){ ?>
    Good Morning, Mr(s). <?php echo $_GET['
        nome']; ?>
  <?php
  }else{
    echo "Good Morning, User!";
  }
  ?>
  </body>
</html>
```

Q7.: Para que serve a instrução na linha 3 do código incluído antes?

☐ Para evitar que esta página seja vulnerável a ataque de *Cross Site Scripting* (XSS).
☐ Para *desligar* a proteção contra XSS disponibilizada pelo próprio *browser*.
☐ Para fazer um ataque de XSS refletido no site do colega.

Tarefa 9 *Task 9*

Engendre e envie ao(à) seu(ua) colega um URL da página acima indicada que, após ser clicado (i.e., ativado) faz com que se abra uma janela de alerta no computador do(a) vitima a dizer

```
Se clicar em OK, a pasta Os Meus Documentos
será eliminada.  Se não quiser que isso
aconteça, afaste-se do computador e bata os
braços como uma galinha.
```

Tarefa 10 *Task 10*

Repita o que fez na tarefa anterior, mas desta vez force o *browser* da vítima a abrir uma janela nova com o texto referido antes no conteúdo. Faça com que a janela tenha foco automaticamente.

Q8.: Em ataques de XSS por reflexão, em que lado da comunicação é executado o código malicioso?

☐ Do lado do cliente. ☐ Do lado do servidor.
☐ Em ambos os lados. ☐ Em lado nenhum.
☐ No meio dos dois.

Tarefa 11 *Task 11*

Os trechos de código incluídos em baixo referem-se à implementação, direta e muito simplista, de um livro de visitas (*guestbook*) recorrendo a PHP+MySQL. Crie os ficheiros `guestbook.php` e `guestbook2.php` e cole lá o código dos dois blocos seguintes.

`guestbook.php`

```
Please, sign my guestbook:
<form action="guestbook2.php" method="post">
Name: <input type="text" name="guestname"><br>
Your comment: <input type="text" name="comment
    "><br>
<input type="submit"></form><br>
<table border=1>
 <tr>
   <td>
   <b>Name</b>
   </td>
   <td>
   <b>Comment</b>
   </td>
 </tr>
 <?php
   $link = mysqli_connect("localhost", "root",
           "mastersofenginf") or
           die("Could not connect");
   mysqli_select_db($link, "Guestbook") or
           die("Could not select database");
   $query  = "SELECT * FROM Guestsign";
   $result = mysqli_query($link, $query) or
           die ("Query failed");
   $signnum = mysqli_num_rows($result);
   for ($i=1 ; $i <= $signnum ; $i++){
     $row_array = mysqli_fetch_array($result,
     MYSQLI_ASSOC);
 ?>
 <tr>
   <td>
   <?php echo $row_array["name"]; ?>
   </td>
   <td>
   <?php echo $row_array["comment"]; ?>
   </td>
 </tr>
 <?php }
 mysqli_close($link);
 ?>
</table>
```

`guestbook2.php`

```
<?
 $link = mysqli_connect("localhost", "root",
         "mastersofenginf") or
         die ("Could not connect");
 mysqli_select_db ($link, "Guestbook") or
         die ("Could not select database");
 $guestname = mysqli_real_escape_string($link,
              $_POST['guestname']);
 $comment   = mysqli_real_escape_string($link,
              $_POST['comment']);
 $query = "INSERT INTO Guestsign(name,comment)
          VALUES('$guestname','$comment')";
 mysqli_query ($link, $query) or
          die ("Query failed: $query<br>" .
          mysqli_error());
  mysqli_close($link);
?>
Obrigado por assinar o livro de visitas.
```

Note que a base de dados de suporte a esta pequena aplicação deve ter um esquema semelhante ao que é refletido pelo *script* seguinte:

```
CREATE DATABASE Guestbook;
USE Guestbook;
CREATE TABLE Guestsign(
   id INT not null AUTO_INCREMENT,
   name TEXT not null, comment TEXT not null,
   PRIMARY KEY (id), INDEX (id), UNIQUE (id));
```

Q9.: A aplicação mencionada nesta tarefa é suscetível a que tipo de ataque?

☐ Ataque de dicionário.
☐ Ataque de pânico.
☐ Ataque de XSS por reflexão.
☐ Ataque de XSS por armazenamento.
☐ XSS baseado em DOM.

Tarefa 12 *Task 12*

Aceda ao livro de visitas de um(a) colega seu(ua) e deixe lá uma mensagem do tipo *ovo kinder*. I.e., deixe uma mensagem com *surpresa*. Experimente, e.g., deixar o seguinte pedaço de código como mensagem:

```
<script type="text/javascript">
alert('You have been compromised!');
</script>
```

Q10.: Como se resolve o problema explorado nesta tarefa?

☐ Chamando a polícia.
☐ Saneando os *inputs* que chegam ao PHP através das caixas de texto.
☐ Não guardando nada que venha do utilizador.
☐ Através da função `$input = mysqli_real_escape_string($input).`
☐ Formatando os *outputs* para algo inócuo à interpretação.
☐ Este problema não pode ser resolvido.

Q11.: Em ataques de XSS por armazenamento, em que lado da comunicação é executado o código malicioso?

☐ Do lado do cliente. ☐ Do lado do servidor.
☐ Em ambos os lados. ☐ Em lado nenhum.
☐ No meio dos dois.