**第十七章 PHP MYSQL身份验证**

**1. 识别访问者**

**2. 实现 简单的访问控制**

HTML表单，要求访问者提供访问站点所需的用户名和密码



当用户输入了不正确的信息时，将给出一个出错信息



当用户提交正确的详细信息时，脚本将显示内容



**secret.php——提供简单的身份验证机制的PHP和HTML**

**————————————————————————————————**

**<?php**

**//create short names for variables**

**$name = $\_POST['name'];**

**$password = $\_POST['password'];**

**if ((!isset($name)) || (!isset($password))) {**

**//Visitor needs to enter a name and password**

**?>**

**<h1>Please Log In</h1>**

**<p>This page is secret.</p>**

**<form method="post" action="secret.php">**

**<p>Username: <input type="text" name="**name**"></p>**

**<p>Password: <input type="password" name="**password**"></p>**

**<p><input type="submit" name="submit" value="Log In"></p>**

**</form>**

**<?php**

**} else if(($name=="user") && ($password=="pass")) {**

**// visitor's name and password combination are correct**

**echo "<h1>Here it is!</h1>**

**<p>I bet you are glad you can see this secret page.</p>";**

**} else {**

**// visitor's name and password combination are not correct**

**echo "<h1>Go Away!</h1>**

**<p>You are not authorized to use this resource.</p>";**

**}**

**?>**

**————————————————————————————————**

**存在的问题：**

**1）该脚本只对一个用户名和密码进行了硬编码**

**2）将密码以普通文本形式保存**

**3）只保护一个页面**

**4）以普通文本形式传输密码**

**2. 保存密码**

**secretdb.php——使用mysql来改进原有的身份验证机制**

**————————————————————————————————**

**<?php**

**$name = $\_POST['name'];**

**$password = $\_POST['password'];**

**if ((!isset($name)) || (!isset($password))) {**

**//Visitor needs to enter a name and password**

**?>**

**<h1>Please Log In</h1>**

**<p>This page is secret.</p>**

**<form method="post" action="secretdb.php">**

**<p>Username: <input type="text" name="name"></p>**

**<p>Password: <input type="password" name="password"></p>**

**<p><input type="submit" name="submit" value="Log In"></p>**

**</form>**

**<?php**

**} else {**

**// connect to mysql 参数分别是 -h -u -p**

**@ $mysql = mysqli\_connect("localhost", "webauth", "webauth");**

**if(!$mysql) {**

**echo "Cannot connect to database.";**

**exit;**

**}**

**// select the appropriate database**

**$selected = mysqli\_select\_db($mysql, "auth");**

**if(!$selected) {**

**echo "Cannot select database.";**

**exit;**

**}**

**// query the database to see if there is a record which matches**

**$query = "select count(\*) from authorised\_users where**

**name = '".$name."' and**

**password = sha1('".$password."')";**

**$result = mysqli\_query($mysql, $query);**

**if(!$result) {**

**echo "Cannot run query.";**

**exit;**

**}**

**$row = mysqli\_fetch\_row($result);**

**$count = $row[0];**

**if ($count > 0) {**

**// visitor's name and password combination are correct**

**echo "<h1>Here it is!</h1>**

**<p>I bet you are glad you can see this secret page.</p>";**

**} else {**

**// visitor's name and password combination are not correct**

**echo "<h1>Go Away!</h1>**

**<p>You are not authorized to use this resource.</p>";**

**}**

**}**

**?>**

**————————————————————————————————**

**数据库可以通过mysql 的root用户的身份连接mysql并运行如下脚本来创建**

**createauthdb.sql——这些mysql查询语句，将创建auth数据库、auth表和两个示例用户**

**————————————————————————————————**

**create database auth;**

**use auth;**

**create table authorised\_users ( name varchar(20),**

**password varchar(40),**

**primary key(name)**

**);**

**insert into authorised\_users values ( 'username',**

**'password' );**

**insert into authorised\_users values ( 'testuser',**

**sha1('password') );**

**grant select on auth.\***

**to 'webauth@localhost'**

**identified by 'webauth';**

**flush privileges;**

**————————————————————————————————**

**2.2 密码的加密**

**PHP提供了一个功能强大的单向加密哈希函数 shal(),返回一个40个字符的伪随机字符串，输出是确定的，即，相同的字符串，每次运行返回的结果相同。**

**<?php**

**echo sha1('password');**

**?>**



**if(($name=="user") && ($password=="pass")) {**

**可以改写为：**

**shal($password) == '5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8'**

**mysql的shal()函数与php的shal()函数，作用相同。**

**$query = "select count(\*) from authorised\_users where**

**name = '".$name."' and**

**password = sha1('".$password."')";**

**2.3 保护多个网页**

**解决办法：HTTP基本身份验证和会话控制。**

**3.使用基本身份验证**

**浏览器可以保存身份验证详细信息，自动发送给web服务器，无须用户介入。**

**HTTP这个特性叫做基本身份验证。HTTP1.1包含一种更安全的方法，叫做 摘要身份验证，它使用哈希算法（MD5）掩饰事务处理的细节。**

**4. 在PHP中使用基本身份验证（自学）**

**http.php——PHP可以触发HTTP基本身份验证**

**———————————————————————————————**

**<?php**

**// if we are using IIS, we need to set**

**// $\_SERVER['PHP\_AUTH\_USER'] and**

**// $\_SERVER['PHP\_AUTH\_PW']**

**if ((substr($\_SERVER['SERVER\_SOFTWARE'], 0, 9) == 'Microsoft') &&**

**(!isset($\_SERVER['PHP\_AUTH\_USER'])) &&**

**(!isset($\_SERVER['PHP\_AUTH\_PW'])) &&**

**(substr($\_SERVER['HTTP\_AUTHORIZATION'], 0, 6) == 'Basic ')**

**) {**

**list($\_SERVER['PHP\_AUTH\_USER'], $\_SERVER['PHP\_AUTH\_PW']) =**

**explode(':', base64\_decode(substr($\_SERVER['HTTP\_AUTHORIZATION'], 6)));**

**}**

**// Replace this if statement with a database query or similar**

**if (($\_SERVER['PHP\_AUTH\_USER'] != 'user') ||**

**($\_SERVER['PHP\_AUTH\_PW'] != 'pass')) {**

**// visitor has not yet given details, or their**

**// name and password combination are not correct**

**header('WWW-Authenticate: Basic realm="Realm-Name"');**

**if (substr($\_SERVER['SERVER\_SOFTWARE'], 0, 9) == 'Microsoft') {**

**header('Status: 401 Unauthorized');**

**} else {**

**header('HTTP/1.0 401 Unauthorized');**

**}**

**echo "<h1>Go Away!</h1>**

**<p>You are not authorized to view this resource.</p>";**

**} else {**

**// visitor has provided correct details**

**echo "<h1>Here it is!</h1>**

**<p>I bet you are glad you can see this secret page.</p>";**

**}**

**?>**

**——————————————————————————————————**