网络与系统安全

实验三 SQL 注入

CONTENTS

目录

实验目的

- 理解SQL注入的原理;
- 学习手工注入的过程;
- 掌握SQL注入工具的使用;
- 掌握SQL注入的防御技术。

实验任务

本次实验创建了一个易受SQL注入攻击的Web应用程序。该Web应用程序包括许多Web开发人员犯的常见错误。我们的目标是找到利用SQL注入漏洞的方法，理解攻击可能造成的损害，并掌握可以帮助防御此类攻击的技术。

1、熟悉MySQL语句

2、sqlmap工具的使用

3、SELECT 语句下的注入攻击

4、UPDATE 语句下的注入攻击

5、防御技术---预处理方式

实验原理

◆SQL语句

插入语句： **insert into** student(id, name, major, grade) **value**(20188197, ‘test’, ‘cs’, 2018);

查询语句： **select** * **from** student **where** id = 20188197;

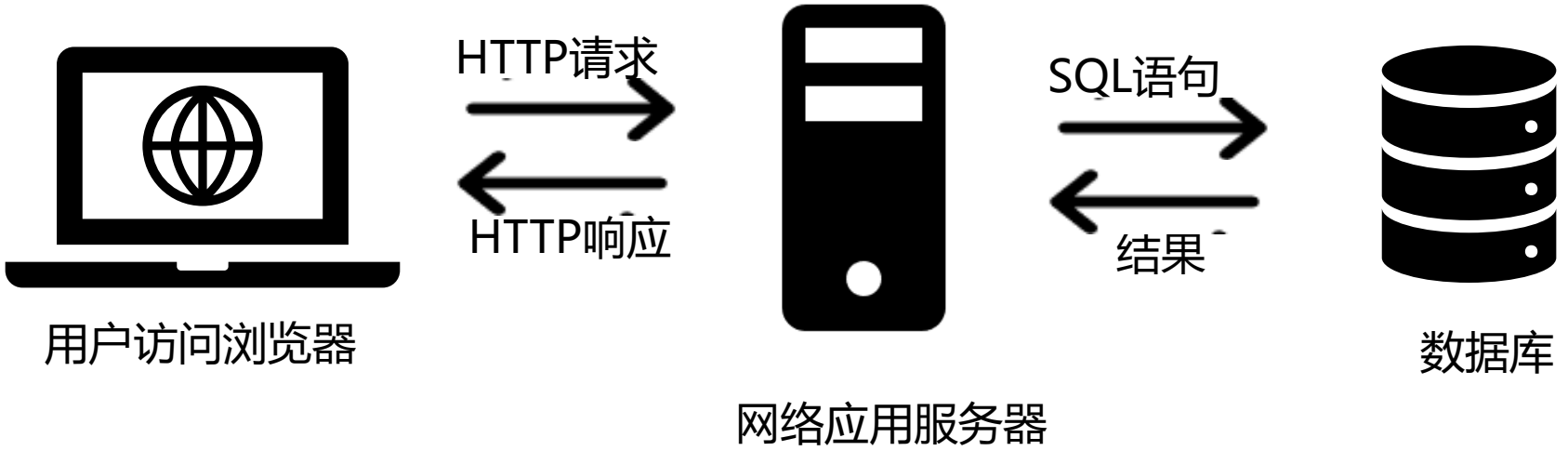
更新语句： **update** student **set** major = ‘cs’ **where** id = 20188197;

删除语句：**delete from** student **where** id =20188197;

实验原理

◆SQL注入产生的原因

当web应用向后台数据库传递SQL语句操作数据库时，如果对用户输入的参数没有经过**严格的过滤处理**，那么攻击者就可以**构造特殊的SQL语句**，**修改原有SQL语句的结构**，从而获取或者修改数据库中的数据。



实验原理

◆SQL注入漏洞原理

SQL注入的本质是把用户输入的数据当做代码来执行，违背了“数据与代码分离”的原则。

1、程序是**弱类型**语言；

2、程序中访问数据库的SQL语句使用了**用户传入的参数；**

3、用户可以通过传入的参数**修改原SQL语句的结构。**

实验原理

◆SQL注入漏洞原理

1、程序是**弱类型**语言；

2、程序中访问数据库的SQL语句使用了**用户传入的参数；**

3、用户可以通过传入的参数**修改原SQL语句的结构。**

```php
<?php
session_start();
// if the session is new extract the username password from the GET request
$input_uname = $_GET['username'];
$input_pwd = $_GET['Password'];
$hashed_pwd = sha1($input_pwd);
```

弱类型语言

用户输入参数

```php
// create a connection
$conn = getDB();
// Sql query to authenticate the user
$sql = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
FROM credential
WHERE name= '$input_uname' and Password='$hashed_pwd';
```

可以通过传入的参数修改SQL语句结构

只有敲代码才能感受到温暖

实验原理

◆SQL注入漏洞原理

```
// Sql query to authenticate the user
$sql = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
FROM credential
WHERE name= '$input_uname' and Password='$hashed_pwd'";
```

希望的用户输入：
用户名：admin
密码：　!QAZ2wsx

实际的用户输入：
用户名：admin’#
密码：

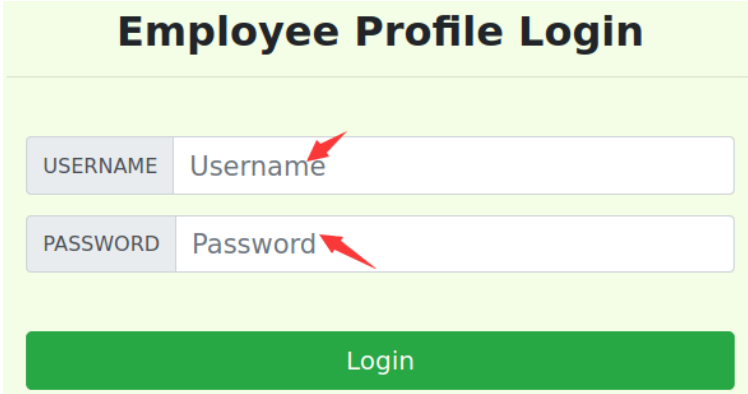
Select * from credential where name=‘admin’ and password=‘!QAZ2wsx’

Select * from credential where name=‘admin’ # and password= ’

实验原理

◆SQL注入示例

示例1，**select**字符型注入：



- 通过用户提供的参数来查询表中的数据

  select * **from** USERS **where** name= ‘admin’ and passwd= ‘!QAZ2wsx’

- 其中name和passwd这两个参数来自于用户的输入：
  - 参数未经验证或编码
  - 黑客输入：’ or 1=1 # 或者 admin’ #

- 应用程序构造查询语句：

  select * **from** USERS **where** name= ‘’ or 1=1 # and passwd= ‘’ ‘

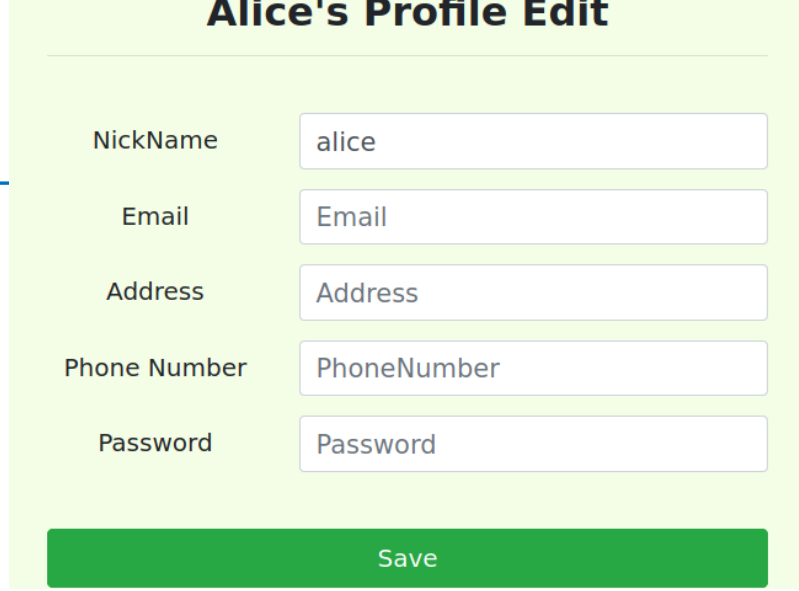
  select * **from** USERS **where** name= ‘admin’ # and passwd= ‘’ ‘

  虽然没有输入密码，依然会有结果返回。

实验原理

◆SQL注入示例

示例2，**update** 字符型注入：



- 通过用户提供的参数来查询表中的数据

update USERS **set** name= ‘alice’ ,email = ‘xx@163.com’ ,… **where** id= ‘123’

- 其中name,email等参数来自用户输入，参数未经验证或编码，黑客在任意参数位置可更改某个用户的工资：
  - NickName输入更改alice的工资：alice’ ，salary=99999 where name = ‘Alice’ #
  - Phone Number输入更改Boby的工资：6503 ‘，salary=100 where name = ‘Boby’ #
- 应用程序构造查询语句：

update USERS **set** name= ‘alice’ ，salary=99999 where name = ‘Alice’ #‘ ,email= ‘xx@163.com’ ,… **where** id= ‘123’

登录自己的用户编辑页面可以更改任意人的信息。

实验原理

◆Sqlmap注入工具

sqlmap是一个使用python语言开发的开源的**渗透测试工具**，可以用来进行**自动化检测**，利用 SQL 注入漏洞，获取数据库服务器的权限。它具有功能强大的**检测引擎**，可以针对各种不同类型数据库的进行**渗透测试**，包括获取数据库中存储的数据，访问操作系统文件等。

Sqlmap工具安装命令：
linux下ubuntu环境：sudo apt-get install sqlmap

Windows下可参考如下链接：
https://blog.csdn.net/qq_46700234/article/details/122906546

实验原理

◆ Sqlmap注入工具

检测某个url的注入点：sqlmap –u “url” 或者 python sqlmap.py –u “url”

获取数据名： --DBS
指定数据库名: -D 数据库名
获取表名： --tables
指定表名： -T
获取字段名： --columns
获取数据： --dump

其他命令可查看帮助：sqlmap -h 或者 sqlmap -hh

示例：将某个表中的所有信息dump下来
sqlmap -u "http://www.seed-server.com/unsafe_home.php?username=admin&Password=" -D 库名 -T 表名 --dump

实验原理

◆ 防御SQL注入的措施

➢ 过滤掉代码

编码前： aaa’ or 1=1#

编码后： aaa\’ or 1=1#

PHP的mysqli有一个内置方法，mysqli::real_escape_string()，它可以用于编码SQL中的特殊字符：反斜杠(\)，撇号(’)百分号(%)等。

```
<?php
  $conn = new mysqli($dbhost, $dbuser, $dbpass, $dbname);
  $eid = $mysqli->real_escape_string($_GET['EID']);
  $pwd = $mysqli->real_escape_string($_GET['Password']);
  $sql = "SELECT Name, Salary, SSN
          FROM employee
          WHERE eid='$eid' and password='$pwd';
?>
```

➢ 预处理语句

- 预处理SQL语句
- 绑定数据
- 执行与获取结果

```
$stmt = $conn->prepare("SELECT name, local, gender
                        FROM USER_TABLE
                        WHERE id = ? and password = ? ");
$stmt->bind_param("is", $id, $pwd);
$stmt->execute();
$stmt->bind_result($bind_name, $bind_local, $bind_gender);
$stmt->fetch();
```

实验原理

◆ 防御SQL注入的措施

➢ 过滤掉代码

过滤或转移字符的方法没有解决根本问题，数据和代码仍然混合了，**不推荐使用。**

```
<?php
  $conn = new mysqli($dbhost, $dbuser, $dbpass, $dbname);
  $eid = $mysqli->real_escape_string($_GET['EID']);
  $pwd = $mysqli->real_escape_string($_GET['Password']);
  $sql = "SELECT Name, Salary, SSN
          FROM employee
          WHERE eid='$eid' and password='$pwd';
?>
```

➢ 预处理语句

使用预处理语句，可信的代码通过代码通道被发送，不可信的用户数据通过数据通道被发送，数据库能够清楚知道代码和数据的界限，**可以防范SQL注入攻击。**

```
$stmt = $conn->prepare("SELECT name, local, gender
                        FROM USER_TABLE
                        WHERE id = ? and password = ? ");
$stmt->bind_param("is", $id, $pwd);
$stmt->execute();
$stmt->bind_result($bind_name, $bind_local, $bind_gender);
$stmt->fetch();
```

实验步骤

1、**熟悉MySQL语句**，查询Alice用户的信息；

2、**sqlmap工具的使用**，把credential表中的信息全部dump下来；

3、**SELECT 语句下的注入攻击**，分别用网页输入信息和curl的方式获取所有用户的信息；尝试输入两条select语句，并分析说明查询结果；

4、**UPDATE 语句下的注入攻击**，分别修改自己的工资，他人的工资和他人的信息；

5、**防御技术---预处理方式**，使用预处理语句机制来修复SQL注入漏洞。

作业要求

提交内容：实验报告（有模板）

截止时间：

实验课后一周内提交至HITsz Grader 作业提交平台，具体截止日期参考平台发布。

- 登录网址：： http://grader.tery.top:8000/#/login
- 推荐浏览器： Chrome
- 初始用户名、密码均为学号，登录后请修改

注意

上传后可自行下载以确认是否正确提交

附录

◆ OWASP

OWASP(Open Web Application Security Project)

http://www.owasp.org

- 一个全志愿者组成的、非营利性机构；
- 开发和出版免费专业开源的文档、工具和标准，如： “The Ten Most Critical Web Application Security Vulnerabilities”， 《A Guide to Building Secure Web Applications》， WebGoat， WebScarab，各种Web代码测试工具等；
- 致力于帮助组织机构理解和提高他们的Web安全；
- 组织各种Web安全会议。

OWASP Top 10: https://owasp.org/www-project-top-ten/

同学们
请开始实验吧！