# 简介

应用程序在向后台数据库传递SQL（Structured Query Language，结构化查询语句）查询时，如果为攻击者提供了影响该查询的能力，就会引发SQL注入。

一个正常的PHP数据查询语句：

$conn = mysql\_connection($host, $username, $password)

$query = “select \* from users where user=’admin’ and passwd=’”

$query = $query.$\_GET[“passwd”].”’”；

$result = mysql\_query($query);

这里查询语句中的passwd是根据用户输入设定的，没有做任何的限定，攻击者可以利用这个修改查询语句。

# 分类

## SQL注入点

字符型

整数型

搜索型：like ‘’

## SQL注入攻击

没有正确的过滤转义字符

错误的类型处理：比如要求传入整数型，最终传入字符型

数据库服务器中的漏洞

SQL盲注：错误没有回显

条件响应：1=1 AND 1=2

条件性差错（select 1/0 from t1 where id=0;）

时间延迟

# 原理

正常的 SQL 语句：

select \* from users where id = 1;//正常情况不会把id写死，而是根据外部传入

动态的构造过程：

$query = "select \* from users where user='admin' and passwd='"

$query = $query.$\_GET["passwd"]."'";

利用动态构造构造注入语句：

不传入id数字，而是传入字符等非法字符

# 测试

简单的规则：发送以外数据来触发异常

识别Web应用的数据输入：

select \* from users where id =1 and 1=1; //TRUE and TRUE，返回正确结果

select \* from users where id =1 and 1=2;//TRUE and FALSE,返回空，SQL注入

了解哪种类型的请求会触发异常

检测服务器响应中的异常

# 利用与防御

## 不同数据库的SQL注入

### Access 的 SQL 注入

Access 注入猜解过程

猜解表名

猜解列名

猜解字段值长度

ASCII 逐字解码法猜解字段值

ASP+Access：

### MySQL 的 SQL 注入

### SQLServer 的 SQL 注入

有趣的出错信息

Group By 与 Having 查询

爆表名与字段名

爆数据记录

## 基于不同方法的条件语句的使用

## 其他SQL注入利用手法

## SQL注入的防御