基于数字校园的SQL注入漏洞防御技术研究

石　怡

（江苏信息职业技术学院，江苏无锡 214153）

【摘要】随着智慧数字校园时代的来临，各高校逐步实现了集网上教学、科研、管理、服务与办公一体化的Web门户平台搭建。校园网为师生员工与校外用户提供了全面、共享与快捷的信息服务，但也面临着网络安全问题的严峻挑战，其中SQL注入攻击的威胁最为常见、最具破坏性。文章分析了校园Web系统所面临的SQL注入漏洞原理，并提出了可行的防御技术方案。

【关键词】校园Web；SQL注入；网络安全；防御技术

【doi:10.3969/j.issn.2095-7661.2021.04.007】

【中图分类号】TP393.08　　【文献标识码】A　　【文章编号】2095-7661(2021)04-0022-04

Research on SQL Injection Vulnerability Defense Technology Based on Digital Campus

SHI Yi

(Jiangsu Vocational College of Information Technology, Wuxi, Jiangsu, China 214153)

Abstract: With the advent of the era of smart digital campus, colleges and universities have gradually realized a web portal platform integrating online teaching, scientific research, management, service and office work. Campus network provides comprehensive, shared and fast information service for teachers, students, employees and users outside the school, but it also faces severe challenges from network security. Among them, SQL injection attack is the most common and destructive threat. This paper analyzes the principles of SQL injection vulnerabilities faced by campus web systems, and proposes feasible defense technical solutions.

Keywords: campus Web; SQL injection; network security; defense technology

SQL注入攻击长期占据Web应用程序安全风险的首位，近年来陆续有高校网站受到SQL注入漏洞的攻击。一方面高校校园网内部子系统众多，功能分散，数据分散，管理难度较大，另一方面部分高校网络安全防御能力不足。被攻击后，可能会造成校园网站的网页信息、学生考试成绩被篡改，学校内部一卡通系统账目被改动，后台隐私数据泄露等安全事件，产生恶劣影响。

Web开发人员在程序设计时如果缺乏网络安全意识就有可能造成安全漏洞隐患[1]。一方面，部分高校网站是由学校内部的纵向科研项目自主研发，参与的老师或学生在安全测试方面的经验不足；另一方面，参与信息系统运维的专业人员也较少，因而加大了高校网站被黑客攻击的风险。要避免SQL注入漏洞产生，Web项目的参与人员都应该具备基本的计算机安全知识，养成良好的安全编程习惯[2]。

1 SQL注入原理分析

SQL注入是一种通过控制输入来篡改SQL语句的执行，攻击后台数据库的技术。其本质是应用程序无法正确判断出用户输入的是数据还是SQL代码。一旦非法数据嵌入到SQL语句后被当作正常代码来执行，数据库服务器就可能遭受破坏。而Web服务器与数据库服务器相互独立，Web服务器往往无法验证回传数据的合法性，进而会将隐私数据反馈给攻击者。

[收稿日期] 2021-10-27

[作者简介] 石怡(1981-)，女，江苏无锡人，江苏信息职业技术学院物联网工程学院讲师，硕士，研究方向：数据库、软件应用开发。

[基金项目] 2021年江苏省高等职业教育高水平专业群建设项目“物联网应用技术专业群”(项目编号:苏教职函[2021]1号)。

1.1 一阶SQL注入

当在Web应用程序的页面中提交单个HTTP请求时，攻击者将包含了某种特定含义的特殊字符，如转义字符、注释字符等封装至SQL命令并完成提交，此时，最容易发生一阶SQL注入攻击。对于设计不完善的应用系统，在处理输入时会忽略合法性的验证。在未经验证的情况下，带有攻击载荷的用户输入被拼接到动态SQL语句中，并提交至数据库执行以获取未授权访问的信息。其执行过程如图1所示。

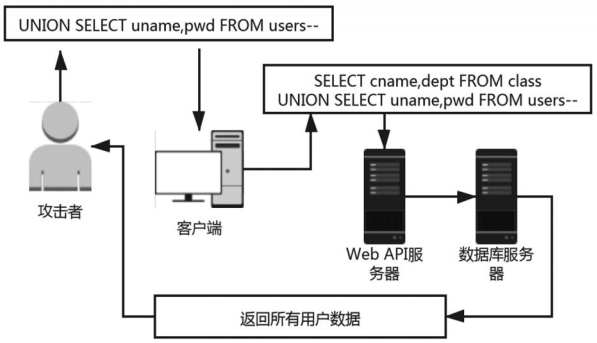

图1　SQL注入执行流程图

在图1场景中，攻击者利用联合查询UNION语法，在正常的SQL语句中拼接了恶意代码，并注释了该语句的后续内容。在获取班级信息的同时也返回得到了所有用户的账号密码信息。

有的高校为节约成本，会将诸如校园门户网站、网上办事大厅这类提供校外网络访问服务的Web应用作为校内纵向项目。参与的师生在Web开发过程中往往更关注于网站功能的实现、如何缩短开发时间，而忽视了安全问题。

1.2 二阶SQL注入

与传统一阶SQL注入相比，二阶SQL注入漏洞通常更隐蔽，有时甚至可以隐藏数年，也较难通过自动扫描工具检测出来[3]。其基本原理是攻击者通过首次注入将恶意数据保存至数据库中，直到再次提交HTTP请求时再将其触发，从而对应用程序构成威胁。

在软件开发过程中，开发人员有时会错误地认为用户输入的数据在经过安全转换后即成为可信任的数据，即已存入数据库的信息都被默认是安全的，执行查询时无需再进行处理。然而程序应用的实现逻辑往往是相互关联的，可能需要经过若干步骤，例如常见的用户注册与用户登录操作。如果对输入数据进行盲目处理，那么一旦在基础代码中与受信任数据相关的其他位置发生了改变，就有可能产生二次漏洞。发生在登录模块常见的二阶注入执行过程如图2所示。

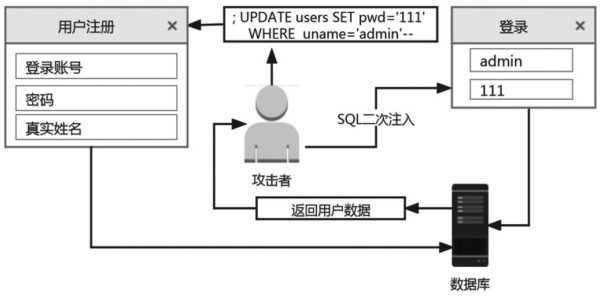

图2　SQL二阶注入执行流程图

在图2场景中，攻击者在用户注册页面提交的第一条SQL语句中插入了UPDATE更新语句，达到了更改数据库admin账户密码的目的。接着在登录页面使用已更新过的密码顺利通过了用户验证。

高校Web网站通常会由行政部门与二级学院自行负责管理。部分管理员仅负责网站的基础运行、资料的上传下载等日常操作，并不具备专业的网络安全知识，在遭遇注入攻击时未必能够及时发现，从而造成校方的损失。

2 SQL注入攻击检测方法

应用系统的源程序代码体现了应用开发人员个人独特的编程风格与技巧，不可避免地会产生漏洞。不管应用程序受到哪种类型的SQL注入攻击，其根本原因是由于开发人员对数据库的理解不够全面，没有意识到代码潜在的安全隐患。要准确识别出SQL注入漏洞特别是二阶注入，可以结合尝试以下几种检测方法：

2.1 复查SQL代码

理解Web应用程序的业务逻辑，复查相关的SQL代码内容，即所有拼接了用户输入的动态SQL语句。对将持久保存在数据库中且能被重用的数据项，还应通过单独操作每项实例，找出并跟踪其执行的后续步骤，以判断是否存在二阶注入。

2.2 检查应用流程

SQL查询中的数据项在未经安全验证的情况下使用时容易引发问题，因此有必要对所有包含多个阶段的应用流程进行检查，例如用户注册、修改账户密码等，以保证提交后的数据能够正确、持久地存在于数据库中。

2.3 处理输入过滤器

如果应用程序中设计了输入过滤器，会对测试输入进行拦截，需要寻找替代语句绕过前台输入过滤器。例如，对于SQL Server数据库，可以使用字符串'adm'+'in'或函数concat ('adm','in')来避开程序对admin表单数据的阻止。如果是Oracle数据库，

可以用chr函数来避开对单引号的阻止。

2.4 检查异常行为

检查所有可能由输入引发的异常行为。除显示使用数据实例的功能之外，还应包括可能存在的隐式使用数据实例的功能，以避免未来导致二次注入的风险。

2.5 验证缺陷数据

对于持久存在于数据库中的缺陷数据，常常会因间接收到攻击而引发异常，例如数据类型转换错误等。针对每个检测出可能存在风险的SQL查询，可采用概念验证方式。尝试为同一输入添加引号来观察异常是否可恢复，或尝试数据库特有的版本标识与字符连接函数等专用结构来确认。

2.6 使用测试工具

有效使用SQL测试工具可提高检测效率。不仅能快速定位到易受攻击的注入点,还具有扩大测试语句的覆盖范围，自动构造SQL盲注代码等优点。最常见的测试工具有Sqlmap、Havij、Safe3 SQL Injector、SQL Poizon等。

3 SQL注入防御有效措施

在掌握了SQL注入的原理与检测方法后，无论是高校网站的开发人员还是运维人员，都可以在代码层与平台层进行一些合理的操作来降低甚至是消除SQL注入的威胁。

3.1 代码层

3.1.1 参数化SQL语句

当前主流的高级语言和数据库访问API都为SQL语句的参数绑定提供了良好的支持。在编码时，SQL语句中需要用户输入的位置可被替换为占位符或变量参数。这样不仅能够优化查询，也能够避免很多在应用中常见的SQL注入问题。在应用程序中应确保始终正确地将参数化语句嵌入到SQL查询的每个可变数据项中，以保证程序执行的安全有效。

以ASP.NET Web应用系统为例，使用ADO.NET提供的SqlClient数据提供程序访问SQL Server数据库，重写根据用户ID判断用户是否存在的参数化代码如下。

```
public bool Exists(int uid)
{
StringBuilder strSql = new StringBuilder();
strSql.Append("select count(1) from users ");
strSql.Append("where uid=@uid");
SqlParameter[] parameters = {new SqlParameter
("@uid", SqlDbType.Int,4)};
parameters[0].Value = uid;
return DbHelperSQL. Exists(strSql. ToString(),
parameters);
}
```

3.1.2 验证输入

应用程序中需要用户输入的位置使用白名单与黑名单相结合的验证方法。白名单设定了允许通过验证的输入规则，用来判断用户数据是否与期望的类型、长度、取值范围或特定格式标准相匹配。既可以是简单的参数类型限制，也可以是正则表达式或较复杂的业务逻辑。为白名单验证辅以黑名单验证。将所有已知的不良字符、字符串或非法模式加入黑名单，在输入验证时如在黑名单中找到该记录则拒绝访问。

3.1.3 设计存储过程

在应用程序中使用预编译后的数据库存储过程，不仅能通过缓存技术提高数据操作性能，还能借助DBMS安全机制实施访问权限控制。用户账户只具有对存储过程的执行权限，而无法直接对数据进行SQL查询，执行数据交互的SQL语句被改写至存储过程中，并被编译为对象存储在数据库中。用户账户只可通过对存储过程的调用来完成与应用程序的交互，从一定程度上限制了攻击者能够访问或修改的数据，减轻了潜在的SQL注入风险。

3.2 平台层

3.2.1 部署Web应用防火墙

Web应用防火墙（WAF）能够为应用程序提供运行时的安全防护策略，有效应对和预防SQL注入[4]。WAF可以是网络硬件设备，也可以是嵌入至Web应用的代码模块。它能够跨越包括网络、Web服务器、应用程序框架以及数据库服务器在内的多层架构。在保持Web基础结构不变的情况下，无缝地处理网络通信，且不耗费服务器资源，实现攻击预防、监控、入侵检测和应用程序加固。高校可采用反向代理机制部署WAF，配置虚拟IP，实现Web 服务器的安全隔离，并设置恰当的防护规则，避免出现错误的WAF 拦截[5]。

3.2.2 加密敏感数据

SQL注入攻击的最终目的是获取对数据库中敏感数据的访问权限，包括高校师生账户密码、银行账号、科研数据等。应对这类重要的敏感信息应该进行安全加密后再存储，而非直接存储信息本身[6]。例如，可在密码口令中加入盐值，即将随机字符串插入口令后再进行HASH算法加密。在系统登录时，将用户在页面输入的信息计算转换为哈希

值后再与数据库中保存的数据进行比较。

3.2.3 加固数据库

通过加固数据库来达到确保数据库安全的目的，可有效降低SQL注入的影响。严格设定用户对数据库使用权限的访问限制方案，将攻击者在设法获取访问权限时可以执行的操作设为最低权限，例如只对存储过程授予EXECUTE许可。高校数据库服务器管理员可定期用最新的修补程序更新数据库，修复旧版本中存在的已知弱点或错误，避免被攻击者利用。

3.2.4 隐藏异常响应提示

避免直接向用户泄露包含与应用程序异常相关的错误明细，例如数据库服务器上的SQL代码错误提示[7]。攻击者能利用这些错误信息来发起SQL注入。通过配置应用框架或Web服务器，尽可能隐藏与应用程序异常产生原因相关的技术信息。页面响应时仅显示自定义的错误信息或是跳转至某个默认页面。

针对ASP.NET Web程序，可在Web.Config配置文件中将customErrors模式设置为On，当响应页面遇到包含500、404等错误状态码时将自动跳转至该指定页面。Java Web程序则可在web.xml配置文件中设置error-page节点下的 error-code与location属性。PHP Web程序可修改php.ini配置文件，将display_errors 设为Off以关闭网页错误提示。

4 结语

随着高校网络数字化的进一步推进，攻击者也在不断地寻找数据库系统的薄弱环节，伺机发起新的SQL注入以非法窃取数据。因此，高校网站的开发与管理者应时刻保持安全意识，始终遵循数据与代码分离的原则，采用规范的编程语句，对用户输入进行安全检查，设计并实施安全合理的网站配置方案，综合应用多种软、硬件技术，尽可能减少SQL注入攻击带来的安全隐患，保障校园Web的安全运行。

【参考文献】

[1]沈子雷.基于Web应用的网络安全漏洞发现与研究[J].无线互联科技,2020(5):19-20.

[2]高深.融媒建设中网络安全问题概述[J].科学技术创新,2020(21):80-81.

[3]刘彦鑫.SQL注入原理及防范方法[J].电子世界,2019(7):178-179.

[4]叶良艳.SQL注入漏洞检测防御关键技术综述[J].安徽电子信息职业技术学院学报,2018(3):19-22.

[5]王乐,王叶静,葛永兴,王唯.Web应用防火墙在高校信息安全中的应用[J].长春师范大学学报,2020(4):80-82,104.

[6]杨龙.校园网Web网站网络安全问题分析与解决方法[J].电脑知识与技术,2016(35):57-58.

[7]吴宗卓.SQL注入防御技术研究与实现[J].电子测试,2015(21):81-82.