存储过程技术在网络考试系统 SQL 注入攻击防御上的应用①

黄龙军

(绍兴文理学院 计算机系, 绍兴 312000)

摘 要: 介绍利用 SQL Server 存储过程来提高网上考试系统的安全性, 讨论 SQL 注入攻击及采用存储过程的方法进行防御, 给出了网上考试系统中随机生成试卷等存储过程的实现方法. 采用存储过程能有效防御 SQL 注入攻击, 提高系统的安全性.

关键词: 存储过程; 安全性; 网上考试; SQL 注入; SQL Server

Application of Stored Procedures to Defense against SQL Injection Attacks in Online Examination System

HUANG Long-Jun

(Department of Computer Science, Shaoxing University, Shaoxing 312000, China)

Abstract: This paper describes the use of stored procedures to improve the security of online examination system. We discuss SQL injection attacks , the method of defense using the stored procedures , and some stored procedures of the online examination system,such as randomly generated papers. Use of stored procedures effectually secures against SQL injection attacks, improves the system security.

Key words: stored procedure; security; online examination; SQL injection; SQL Server

网上考试系统是目前各高校及考试机构进行考试的重要平台, 网上考试系统的安全性至关重要. 但在众多基于 ASP.NET 的网上考试系统中的实现中, 往往如文献 1 所示, 在应用程序的代码中直接嵌入 SQL 语句. 这样的实现方式容易导致 SQL 注入攻击[2], 从而导致系统安全性差; 另外, 还存在代码冗长、系统可维护性差、网络通信流量大等问题.

1 SQL注入攻击

1.1 SQL 注入攻击概述

SQL(Structured Query Language, 结构化查询语言), 是关系数据库的标准语言[3], 其功能强大, 包括数据查询、数据定义、数据操纵、数据控制等多方面的功能. SQL 本身并没有攻击漏洞, 但在应用程序的代码中直接嵌入 SQL 语句就容易产生攻击漏洞. SQL 的字符串的界定符是一对单引号, 如'admin'; or 运算符为“或”运算, 表达式 “<表达式 1> or <表达式 2>”中只要表达式 1 或表达式 2 为真, 则整个表达式为真. 恶意攻击者经常利用这些知识进行 SQL 注入攻击. 经过笔者研究发现, 若攻击者采用 SQL 的单行注释符号“--”和单引号进行 SQL 注入攻击, 会对系统的安全性造成致命威胁.

SQL 注入攻击的原理[4]是在客户端提交特殊代码, 非法获取服务器信息. SQL 注入攻击[4]是指攻击者把破坏性的 SOL 命令输入到 Web 页面的表单域, 导致 Web 服务器执行恶意的 SQL 命令, 造成数据库中的数据泄露、更改、破坏. 网上考试系统一旦被 SQL 注入攻击, 损失是难以挽回的.

1.2 SQL 注入攻击分析

简单起见, 我们假设存放网上考试系统的管理员

① 收稿时间:2012-06-14;收到修改稿时间:2012-07-11

用户信息的表(tbluser)的结构如表 1 所示.

表 1 管理员表(tbluser)

字段名	数据类型	描述	备注
ID	bigint	编号	主键，自增
UserName	nvarchar(10)	用户名	候选键
Password	varchar(20)	密码	

1.2.1 利用用户登录漏洞

在 C#代码中采用嵌入式 SQL 编程时，实现用户登录验证的 SQL 操作串通常如下:

string strSql = "select * from tbluser where username ='" + txtUid.Text + "' and Password='" + txtPwd.Text +"'";

如果攻击者在用户名文本框(txtUid)中输入“' or 1=1--”，则 strSql 变为“select * from tbluser where username ='' or 1=1--' and Password=''”，因为“1=1”是永真条件，“--”之后是被 DBMS 视为空白的注释，那么只要管理员表有数据，登录就能通过验证，非法用户以管理员的身份登录系统，对于网上考试系统而言，所造成灾难是致命的.

如果攻击者在用户名文本框(txtUid)中随意输入“test”，在密码文本框(txtPwd)中输入“' or 1=1--”，strSql 变为“select * from tbluser where username ='test' and Password='' or 1=1--'”，因为“1=1”是永真条件，只要用户表有数据，登录也能通过验证.

如果攻击者在用户名文本框(txtUid)中输入“test”，在密码文本框(txtPwd)中输入“' or '1'='1”，strSql 变为“select * from tbluser where username ='test' and Password='' or '1'='1'”，因为“'1'='1'”是永真条件，只要用户表有数据，登录亦能通过验证.

1.2.2 利用查询输入漏洞

在 C#代码中采用嵌入式 SQL 编程时，实现用户模糊查询的 SQL 操作串通常如下:

string strSql = string.Format("select * from tbluser where username like '%{0}%'", txtUid.Text);

如果攻击者在用户名文本框(txtUid)中输入“' drop table users--”，则 strSql 变为“select * from tbluser where username like '%' drop table users--%'”，这个 SQL 语句的执行将把用户表直接删掉！SQL 注入攻击的破坏性可想而知.

2 利用存储过程防御SQL注入攻击

符凤平[4]对 Web 网站安全技术进行了分析. 下面我们讨论利用存储过程的方法来防御 SQL 注入攻击，从而提高网上考试系统的安全性，避免数据泄露、更改、破坏.

2.1 存储过程的优点

存储过程是预先编译好的一组 Transact-SQL 语句，作为一个单元存储在服务器端并在服务器端执行. 存储过程可以带输入、输出参数，具有返回值. 存储过程的优点[3]如下:

(1) 存储过程可以强制应用程序的安全性，可以防御 SQL 注入攻击.

(2) 存储过程大大减少网络通信流量. 客户端应用程序调用存储过程时，只需通过网络发送存储过程名和参数，就可以在数据库服务器执行该存储过程，执行完成后，返回给客户端应用程序结果状态或将最终结果集，无需通过网络传送大量的 SQL 操作命令和中间结果，最大限度地减少网络通信流量.

(3) 存储过程在服务器上注册，执行效率高.

(4) 存储过程大大提高应用程序的可维护性和设计效率. 存储过程将相关业务逻辑封装在一起，可以大大提高整个软件系统的可维护性. 存储过程在服务器端创建好后就可以在应用程序中任意调用，大大提高了应用程序的设计效率.

2.2 SQL 注入攻击的防御

基于存储过程的优点，我们主要采用存储过程的方法防御 SQL 注入攻击. 下面，我们以管理员登录验证存储过程为例来进行说明.

2.2.1 创建存储过程 spLogin

在 SQL Server 2008 中创建存储过程 spLogin，验证用户名和密码的合法性，具体如下:

```
create proc spLogin(@username nvarchar(20), @pwd varchar(20)) with encryption as
begin
  if(select count(*) from tblUser where UserName =@username and password=@pwd)=0
    return 0 --返回 0 表示验证失败
else
  return 1 --返回 1 表示验证成功
end
```

2.2.2 调用存储过程 spLogin

在 Visual Studio 2010(VS 2010)中调用上面的存储过程 spLogin, 具体代码如下:

```
SqlConnection cnn = new SqlConnection(cnnStr);
cnn.Open();
SqlParameter[] paras = {
  new SqlParameter("@username", txtUid.Text),
  new SqlParameter("@pwd", txtPwd.Text),
  new SqlParameter("@res", SqlDbType.Int) };
//指定@res 为返回值参数
paras[2].Direction = ParameterDirection. Return Value;
//以存储过程名创建命令对象
SqlCommand cmd = new SqlCommand("spLogin", cnn);
//指定命令对象类型为存储过程
cmd.CommandType = CommandType. Stored Procedure;
foreach (SqlParameter p in paras) cmd. Parameters.Add(p);//为命令对象添加参数
//调用 ExecuteNonQuery()执行存储过程 spLogin
cmd.ExecuteNonQuery();
if (paras[2].Value.ToString() == "1")
  Response.Write("<script>alert(' 成 功 登 录 !') </script>");
else
  Response.Write("<script>alert(' 用 户 或 密 码 错 误!')</script>");
```

2.2.3 SQL 注入攻击防御分析

采用存储过程的方法实现登录验证功能时, 如果攻击者在用户名文本框(txtUid)中随意输入“test”, 在密码文本框(txtPwd)中输入“' or 1=1--”, 得到的返回值为 0, 验证不能通过, 有效的防御了 SQL 注入攻击. 其主要原因是输入的信息是被参数化了的. 在 SQL 语言中, 字符串中的“''”表示一个单引号“'”, 输入的“' or 1=1--”参数化为一个字符串“'' or 1=1--”传递给参数@pass, 相当于在 SQL Server 2008 中用 exec 执行该存储过程, 具体如下:

```
declare @return_value int
exec @return_value = spAdminLogin @username = 'test',@pwd = ''' or 1=1--'
```

对于其它 SQL 注入攻击的输入, 处理方式类似, 不再赘述. 同时, 我们可以在 C#应用程序中对用户输入的数据进行合法性检查, 对输入的单引号、EXEC、Drop 等可能导致 SQL 注入攻击的字符或字符串进行必要的处理, 从而更有效的防御 SQL 注入攻击. 另外, 可以通过在应用程序中采用异常处理等方法, 避免程序崩溃导致数据库信息的泄漏.

3 网上考试系统的部分存储过程

如上述所示, 利用存储过程的方法能有效防御 SQL 注入攻击, 提高网上考试系统的安全性, 所以在网上考试系统的开发过程中我们编写了大量的存储过程, 应用程序的功能通过调用相应存储过程来实现. 篇幅所限, 我们主要讨论随机抽题(生成某考生试卷)、显示某考生试卷、收集所有考生试卷等存储过程. 在这里, 我们假设考试题型有 0-3 等四种, 每种题型随机抽取三道题目. 涉及到的表有问题表(tblQuestion)、考生表(tblExaminee)、选题表(tblSelect), 表结构如下所示.

tblQuestion(Id, Qid, Description, Type, Used)

各字段含义为: 题目序号, 分类型的编号, 题目内容, 题型, 已用标记. 字段 Id 为主键, 自动增量.

tblExaminee(Id, Number, Name, Password)

各字段含义为: 考生序号, 考号, 考生姓名, 登录密码. 字段 Id 为主键, 自动增量.

tblSelect(Id, Eid, Qid)

各字段含义为: 选题序号, 考生序号, 题目序号. 字段 Id 为主键, 自动增量.

3.1 随机生成试卷

存储过程 spGetQuestionsByType 按问题的类型随机生成一份试卷, 每种题型生成 3 道题; 若每人所抽的题目可以相同, 则选题条件中省去“used=0”; 题数也可以用参数传入. 随机生成题目采用按 newid()函数值排序的方法, 该函数创建 uniqueidentifier 类型的唯一值. newid()每次产生的值都不一样, 根据 newid()的值进行排序, 则每次的结果也是不一样的, 从而实现随机效果. 存储过程的参数@eid 为学生序号.

```
create procedure spGetQuestionsByType(@eid int)
as
begin
  set nocount on;--不返回计数
  declare @type tinyint;--申明循环变量
  set @type=0;--循环变量赋初值
```

```
while @type<=3 --按题型随机抽取道题目
begin
  insert into tblSelect select top 3 @eid, [id] from tblQuestion
    where [type]=@type and used=0
    order by newid();--随机选择题号插入选题表
    set @type=@type+1;--循环变量增值
end
--更新问题表中已用标记
update tblQuestion set used=1 where [id] in (select Qid from tblSelect where Eid=@eid);
end
```

3.2 按考生号显示试卷

存储过程 spShowQuestionsByNumber 显示某考号的考生所抽到的所有题目，参数@number 为考号，可以是考生的准考证号或学号.

```
create procedure spShowQuestionsByNumber (@number varchar(20)) as
begin
  select [id],Qid,Description,[Type]
  from tblQuestion
  where [id] in (
    select Qid from tblSelect
    where Eid = ( select [id] from tblExaminee
      where number=@number))
  order by [Type];
end
```

3.3 试卷收集的存储过程

若要显示所有考生所抽到的题目，即实现考生的试卷收集功能，则可以使用游标逐行获取考生的考号、姓名并执行存储过程 spShowQuestionsByNumber 获取该考生的试卷. 为模块化编程起见，一样可以定义一个存储过程 spGetAllResults，按考号升序显示所有考生所抽到的所有题目. 执行本存储过程可以把所有考生试卷导出到报表或 Excel、Word 等 Office 文档中，以起到收集保存试卷的作用.

```
create procedure spGetAllResults as
begin
  --声明遍历考生表的游标
  declare csExaminee cursor for select Number,Name from tblExaminee order by Number;
  open csExaminee;--打开游标
  --声明变量
  declare @Number varchar(20);
  declare @Name nvarchar(10);
  --从游标提取一个考生的考号和姓名
  fetch next from csExaminee into @Number, @Name;
  --遍历学生表，逐行提取数据
  while @@fetch_status=0
  begin
    --显示考生的考号和姓名
    select @Number Number, @Name Name;
    --显示该考生的选题情况
    exec spShowQuestionsByNumber @Number;
    fetch next from csExaminee into @Number, @Name;
  end
  close csExaminee;--关闭游标
  deallocate csExaminee;--释放游标
end
```

VS 2010 中使用 SqlCommand 对象执行存储过程类似于 2.2 节所示，获得数据可以使用 SqlCommand 对象的 ExecuteReader()等方法，更新数据可以采用 ExecuteNonQuery()方法.

4 结语

我们采用基于存储过程的方法实现网上考试系统，编程过程中综合其它安全控制的方法，如权限控制、Session 对象、数据加密等，能有效的防御了 SQL 注入攻击，提高了网上考试系统的安全性；另外，系统实现的代码得到精简，系统的网络通信流量得到减少，系统的可维护性、设计效率、执行效率等都得到了提高.

参考文献

1 王德安,刘雁南,林大勇.基于 ASP.NET 网上考试系统设计与实现.电脑编程技巧与维护,2012(3):23–27.

2 Watson K.C#2005 数据库编程经典教程.北京:人民邮电出版社,2007.234–265.

3 闪四清,邵明珠.SQL Server 2008 数据库应用实用教程.北京:清华大学出版社,2009.89–256.

4 符凤平.Web 网站安全技术分析.计算机系统应用,2008,17(12):162–165,131.