haby0 / 2019-01-24 08:40:00 / 浏览数 4623 安全技术 WEB安全 顶(0) 踩(0)

前言

PHP数据对象(PDO)扩展为PHP访问数据库定义了一个轻量级的一致接口。PDO提供了一个数据访问抽象层,这意味着,不管使用哪种数据库,都可以使用相同的函数(5.1发行,在PHP

5.0的PECL扩展中也可以使用,无法运行于之前的PHP版本。今天我们讨论PDO多语句执行(堆叠查询)和PDO预处理下的SQL注入问题导致SQL注入的问题。如有不足,7

PDO多语句执行

PHP连接MySQL数据库有三种方式(MySQL、Mysqli、PDO),同时官方对三者也做了列表性比较:

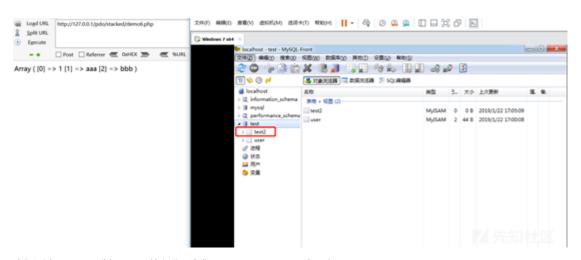
		Mysqli		PDO		MySQL
引入的PHP版本	5.0		5.0		3.0之前	
PHP5.x是否包含	是		是		是	
服务端prepare语句的支持情况	是		是		否	
客户端prepare语句的支持情况	否		是		否	
存储过程支持情况	是		是		否	
多语句执行支持情况	是		大多数		否	

可以看到Mysqli和PDO是都是支持多语句执行的,我们对比一下看看两者的区别

1. Mysqli通过multi_query()函数来进行多语句执行。

```
<?php
$host='192.168.27.61';
$dbName='test';
$user='root';
$pass='root';
$mysqli = mysqli_connect($host,$user,$pass,$dbName);
if(mysqli_connect_errno())
{
    echo mysqli_connect_error();
}
$sql = "select * from user where id=1;";
$sql .= "create table test2 like user";
$mysqli->multi_query($sql);
$data = $mysqli->store_result();
print_r($data->fetch_row());
mysqli_close($mysqli);
```

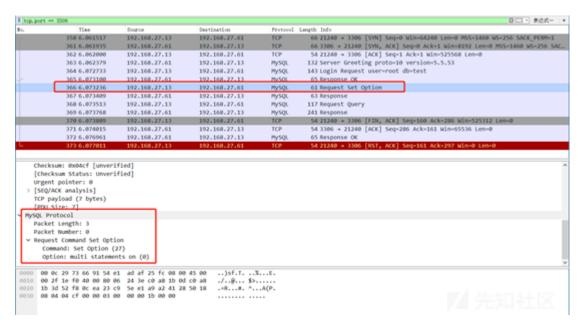
请求脚本后发现数据库中成功创建了test2表,说明多语句成功执行



我们通过wireshark分析一下,首先登录请求Multiple statements字段未设置

```
tqp.port == 3306
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 □□・ 表达式ー
                                                                                                                                                                                                                                      Destination
192.168.27.61
192.168.27.13
                                                                                                                                          192,168,27,13
192,168,27,61
                                                                   14 0.290782
15 0.291636
                                                                                                                                                                                                                                                                                                                                                                                   66 23990 + 3306 [SYN] Seq=0 Min=64240 Len=0 PSS=1460 MS=256 SA
66 3306 + 23990 [SYN, ACK] Seq=0 Ack=1 Min=8192 Len=0 PSS=1460
                                                                                                                                                                                                                                                                                                                                                                                      54 23990 + 3306 [ACK] Seq-1 Ack-1 Min-6553(
132 Server Greeting proto-10 version-5.5.53
14) Login Request user-root db-test
                                                                   16 0.292050
17 0.293132
                                                                                                                                            192.168.27.13
192.168.27.61
                                                                                                                                                                                                                                        192.168.27.61
192.168.27.13
                                                                                                                                                                                                                                                                                                                                                                                107 Cgr. Magazin value - 1001 Cgr. Magazin C
                                                                                                                                                                                                                                                                                                                                   MySQL
MySQL
MySQL
                                                                   19 0.29368
                                                                                                                                             192.168.27.61
                                                                                                                                                                                                                                        192.168.27.13
                                                                                                                                                                                                                                      192.168.27.61
192.168.27.13
192.168.27.61
                                                                   20 0, 293815
                                                                                                                                            192,168,27,13
                                                                   21 0.29404
22 0.29418
                                                                                                                                            192.168.27.61
192.168.27.13
                                                                 23 0.294233
                                                                                                                                        192.168.27.13
                                                                                                                                                                                                                                 192.168.27.61
                                                                                                                                                                                                                                                                                                                                 TCP
                                                                                                                                                                                                                                                                                                                                                                                      54 3306 + 23990 [ACK] Seq-277 Ack-159 Min-65536 Len-0
54 3306 + 23990 [FIN, ACK] Seq-277 Ack-159 Min-65536 Len-0
54 23990 + 3306 [ACK] Seq-159 Ack-278 Min-525568 Len-0
                                                                  25 0.294418
26 0.294448
                                                                                                                                          192.168.27.61
192.168.27.13
                                           Extended Client Capabilities: 0x000a
```

通过multi_query()函数可以看到在执行Query前向Mysql服务器发送了一次Set Option请求将multi statements设置打开



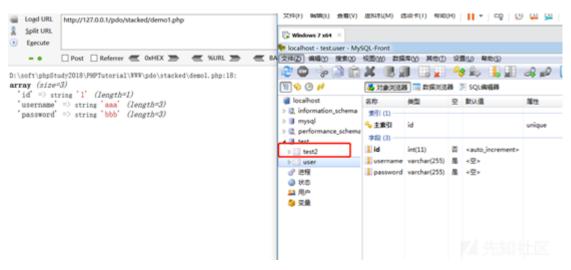
而使用普通的mysqli_query()函数,在执行Query前不会向Mysql服务器发送set option请求

```
A STANKE GER
                 ◆ © 127.00.1/pdo/stacked/de
                                                                                                                                                                                                                                INT - SQL- UNION BASED- ERROR/DG
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              of 2000 (6 2000 - 1)06 [5/8] Sep-8 kin-6220 (ee. 66 3)06 = 2000 [5/8] Sep-8 kin-6220 (ee. 66 3)06 = 2000 [5/8], ACK | Sep-8 kin-6553 132 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 132 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 Server Great (ee. 6000 from 1)0 Ack = 1 kin-6553 133 S
                 Egecute
                                                           □ Post □ Referrer Œ OuHEX 3
 (!) Warning: mysqli_fetch_assoc() exp
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              65 Response OK
117 Request Query
241 Response Error 1864
59 Request Quit
 Call Stack
        # Time
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                54 23990 = 3306 [FIN, ACK] Seq-158 Ack-277 Win-65280 Len-0
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     TCP 54 3306 = 29990 [ACK] Seq=277 Ack=159 Min=65516 Len=0
TCP 54 3306 = 29990 [F18, ACK] Seq=277 Ack=159 Min=65536 Len=0
TCP 54 23990 = 3306 [ACK] Seq=159 Ack=278 Min=525568 Len=0
                                                                                                                                                                                                                                                                                          24 0.294305
25 0.294418
                                                                  0.2047
                                                                                                                                                                                                                                                      Urgent pointer: 0
[StQ/ACK analysis]
TCP psyload (187 bytes)
[POU size: 187]
FSQ. Protocol
Packet Length: 183
Packet Mumber: 1
[From Code: 1864
```

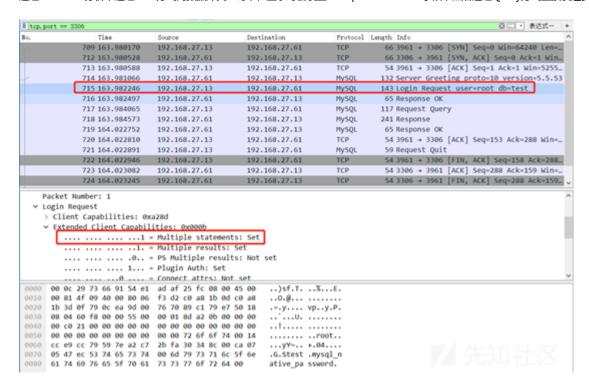
2.使用PDO中的query()函数同数据库交互

```
echo $e;
}
$sql = "select * from user where id=1;";
$sql .= "create table test2 like user";
$stmt = $pdo->query($sql);
while($row=$stmt->fetch(PDO::FETCH_ASSOC))
{
   var_dump($row);
   echo "<br/>;
}
```

请求脚本后发现数据库中成功创建了test2表,说明多语句成功执行



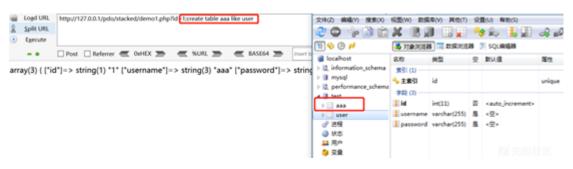
通过wireshark分析,通过PDO方式同数据库交互时,在登录时会设置Multiple statements字段,然后通过Query方式直接发送多语句到Mysql服务器



PDO默认支持多语句查询,如果php版本小于5.5.21或者创建PDO实例时未设置PDO::MYSQL_ATTR_MULTI_STATEMENTS为false时可能会造成堆叠注入

```
}
$id = $_GET['id'];
$sql = "SELECT * from user where id =".$id;
$stmt = $pdo->query($sql);
while($row=$stmt->fetch(PDO::FETCH_ASSOC))
{
   var_dump($row);
   echo "<br/>;
}
```

\$id变量可控,构造链接访问,成功创建aaa数据表



如果想禁止多语句执行,可在创建PDO实例时将PDO::MYSQL_ATTR_MULTI_STATEMENTS设置为false

```
new PDO($dsn, $user, $pass, array( PDO::MYSQL_ATTR_MULTI_STATEMENTS => false))
```

MySQL预处理

MySQL数据库支持预处理,预处理或者说是可传参的语句用来高效的执行重复的语句。

MySQL官方将prepare、execute、deallocate统称为PREPARE STATEMENT

```
预制语句的SQL语法基于三个SQL语句:
prepare stmt_name from preparable_stmt;
execute stmt_name [using @var_name [, @var_name] ...];
{deallocate | drop} prepare stmt_name;
```

PDO预处理

PDO分为模拟预处理和非模拟预处理。

模拟预处理是防止某些数据库不支持预处理而设置的,在初始化PDO驱动时,可以设置一项参数,PDO::ATTR_EMULATE_PREPARES,作用是打开模拟预处理(true)或者关手模拟预处理则是通过数据库服务器来进行预处理动作,主要分为两步:第一步是prepare阶段,发送SQL语句模板到数据库服务器;第二步通过execute()函数发送占位符

首先我们通过wireshark抓包方式对比一下模拟预处理和非模拟预处理模拟预处理代码:

```
<?php
$dbms='mysql';
$host='192.168.27.61';
$dbName='test';
$user='root';
$pass='root';
$dsn="$dbms:host=$host;dbname=$dbName";
    $pdo = new PDO($dsn, $user, $pass, array( PDO::MYSQL_ATTR_MULTI_STATEMENTS => false));
} catch (PDOException $e) {
   echo $e;
$username = $_GET['username'];
$sql = "select * from user where username = ?";
$stmt = $pdo->prepare($sql);
$stmt->bindParam(1,$username);
$stmt->execute();
while($row=$stmt->fetch(PDO::FETCH_ASSOC))
   var_dump($row);
   echo "<br>";
```

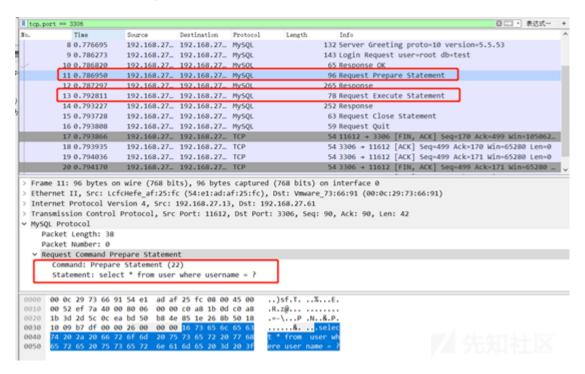
PDO在模拟预处理通过wireshark抓包可以看到是将处理完的SQL语句发送给MySQL服务器

```
192.168.27_ 192.168.27_ MySQL
                                                                              143 Login Request user-root db-test
           7 0.064830
                          192.168.27_ 192.168.27_ MySQI
                                                                              65 Response OK
                          192.168.27_ 192.168.27_ MySQL
192.168.27_ 192.168.27_ MySQL
192.168.27_ 192.168.27_ MySQL
          9 0.066025
                                                                              102 Request Query
          18 8.867821
                                                                              248 Response
          11 0.067343
                                                                               59 Request Quit
          12 0.067397
                          192.168.27_ 192.168.27_ TCP
                                                                               54 11053 - 3306 [FIN, ACK] Seq=143 Ack=284 Win=1050880 L
          13 0.067490
                          192.168.27_ 192.168.27_
                                                                               54 3306 + 11053 [ACK] Seq=284 Ack=144 Win=65536 Len
                                                                              54 3306 → 11053 [FIN, ACK] Seq=284 Ack=144 Win=65536 Len
          14 0.067596
                          192.168.27_ 192.168.27_ TCP
                                                                               54 11053 - 3306 [ACK] Seg=144 Ack=285 Win=1050880 Len=0
          15 0.067623
                          192.168.27_ 192.168.27_ TCP
 Internet Protocol Version 4, Src: 192.168.27.13, Dst: 192.168.27.61
  Transmission Control Protocol, Src Port: 11053, Dst Port: 3306, Seq: 90, Ack: 90, Len: 48
 MySQL Protocol
    Packet Length: 44
    Packet Number: 0

→ Request Command Query

       Command: Query (3)
       Statement: select * from user where username
      00 Oc 29 73 66 91 54 e1
                                  ad af 25 fc 08 00 45 00
                                                              .x.v@......16-...P
      00 58 ef 56 40 00 80 06 00 00 c0 a8 1b 0d c0 a8
      1b 3d 2b 2d 0c ea 0d 13 06 6c 36 2d cf e4 50 18
      10 09 b7 e5 00 00 2c 00 00 00 03 73 65 6c
74 20 2a 20 66 72 6f 6d 20 75 73 65 72 20
0030
0050
```

非模拟预处理代码,在\$username = \$_GET['username'];代码前增加\$pdo->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);这里就是上面提到的,首先给MySQL服务器发送SQL语句模板,然后通过EXECUTE发送占位符参数给服务器

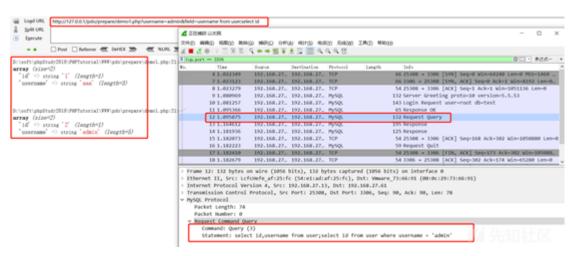


预处理下的安全问题

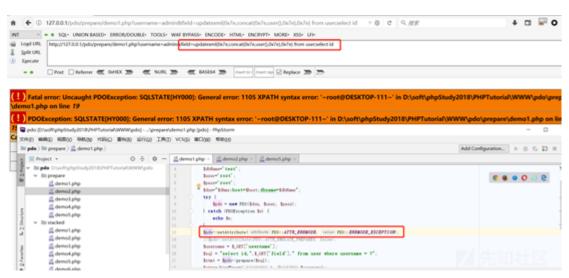
模拟预处理下

```
$stmt->bindParam(1,$username);
$stmt->execute();
while($row=$stmt->fetch(PDO::FETCH_ASSOC))
{
   var_dump($row);
   echo "<br/>;
}
```

可以看到sql语句field字段可控,这样我们构造field,达到多语句执行的效果。

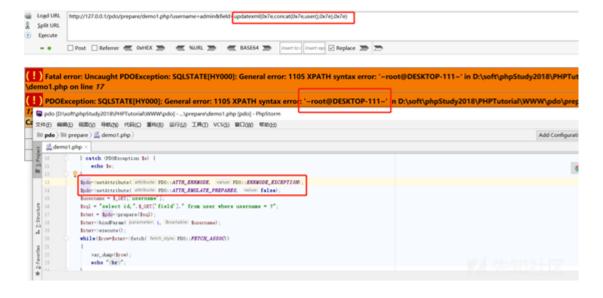


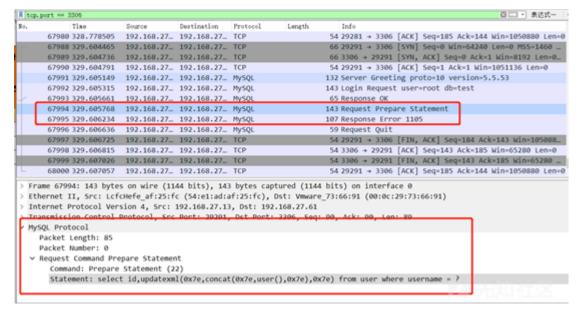
当设置\$pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);时,也可以达到报错注入效果



将上面模拟预处理代码\$pdo->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);的注释关闭来进行非模拟预处理

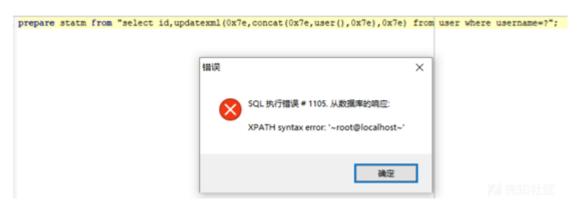
同样的field字段可控,这时多语句不可执行,但是当设置\$pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);时,也可进行报错注入





这里可进行报错注入是因为MySQL服务端prepare时报错,然后通过设置PDO::ATTR_ERRMODE将MySQL错误信息打印在MySQL中执行prepare语句

prepare statm from "select id,updatexml(0x7e,concat(0x7e,user(),0x7e),0x7e) from user where username=?";



总结

- 1. 使用PDO时尽量使用非模拟预处理。
- 2. 创建PDO实例时将PDO::MYSQL_ATTR_MULTI_STATEMENTS设置为false,禁止多语句查询。
- 3. SQL语句模板不使用变量动态拼接生成

参考

https://dev.mysql.com/doc/apis-php/en/apis-php-mysqli.quickstart.multiple-statement.html https://secure.php.net/manual/en/ref.pdo-mysql.php#pdo.constants.mysql-attr-multi-statements https://www.leavesongs.com/PENETRATION/thinkphp5-in-sqlinjection.html

点击收藏 | 1 关注 | 1

上一篇: Asset Discovery: ... 下一篇: 保障IDC安全: 分布式HIDS集群...

- 1. 0 条回复
 - 动动手指,沙发就是你的了!

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

<u>社区小黑板</u>

目录

RSS <u>关于社区</u> 友情链接 社区小黑板